

SPECTRUM

VILÁG 21.

+4 oldal **ENTERPRISE**

49 Ft



A következő **Spectrum 48K** (S151-S158, S160) és **Spectrum 128K** (S159) kollekciók is megrendelhetők utánvétellel a **Spectrum Világ** címén (Budapest, Pf.: 363, 1519) keresztül. Egy kollekció ára ÁFA-val és postaköltséggel együtt **300,- Ft.** Megrendeléseikben kérjük a kollekciók sorszámát pontosan megnevezni!

S151-A: Abracadabra I-II / The Duct / Habilit / Dea Tenebrarum / Swat!
B: Robotscape / Circus Games (part 1-6)

S152-A: Vampire's Empire (+ part 3) / Technocop (+ part 3) / Echelon (part 1-3) / Fire & Forget
B: Hellfire Attack (part 1-7) / Shoot Land (+ part 4) / Shoot Out / Mad Mix, the Pepsi Challenge

S153-A: Captain Blood / Task Force / Robocop (+ part 4) / Colditz / Go into Second Gear / Manhattan
B: Aquarius / Carrots from Space / Sheep Dog / Norman Castle / Jewels of Darkness I-II-III / Estimator Racer / Soft & Cuddly

S154-A: Chicago 30's / Arctic Fox / Championship Sprint (+ editor) / Cavern Fighter
B: Bear A Grudge / Corn Cropper / Fire On the Water / Dreadnoughts / Gerry the Germ / Streethawk

S155-A: Turbo Boat Simulator / Exploding Fist + / Rally Simulator / Motor Massacre (+ part 3) / Tuareg
B: Death Stalker / International Rugby Simulator / Jocky Wilson's Darts Challenge / Gi Hero / Peter Pack Rat / Traz

S156-A: Vindicator (part 1-3) / Return of the Jedi (Domark) / Pacmania / Bounty Bob Strikes Back
B: Four Soccer Simulators (11 A Side, Indoor Soccer, Street Soccer, Soccer Skills) / Strippoker 2

S157-A: Roy of the Rovers / Artura / Sabrina / Chubby Gristle / Secret Mission
B: Rock'n Roller / Wells Fargo / Lightning Simulator / Free Climbing (part 1) / Professional Skateboard Simulator / River Raid / Repulsar

S158-A: Soapland (level 2 - kéri a level 1-ből kimentett állást) / Foxo Fights Back / Blade the Warrior / Babaliba / Gyron Atrium / The Rocky Horror Show
B: Skooldaze / Rocky / Strontium Dog the Killing / Engineer Humpty / Wormy Wawe / Chiller / Yacht Race

S159 (128K) - A: The Raven 128 / Ghostbusters 128 / The Pawn 128
B: Stalker 128 / Tank 128 / Bionic Commando 128

S160-A: Sam Stoat / Jet Set Willy II. / The Saga of Erik the Viking / Witch's Cauldron / Gremlins the Adventure / Airwolf
B: Mr. Mouse / Mc. Wash / Sheer Panic / 3D Star Wars / Race Fun / Speed Duel / Joust! (LOAD™ CODE) / Androids / Kong II. (Strikes Back) / Fighting Warrior / QS-Chess

KAISER PÉTER, budapesti Olvasónk küldte be ezt a szép borítótervet. Sajnos ilyen – előre meg nem rendelt – munkáért pénzbeli honoráriumot küldeni nem tudunk, ám a munka minősége alapján feltétlenül úgy gondoltuk, hogy közöljük a **SpV.**-ban. Péter, csak így tovább!



MINDEN KEDVES OLVASÓNKNAK KELLEMES KARÁCSONYI ÜNNEPEKET, ÉS EREDMÉNYEKBEN GAZDAG BOLDOG ÚJ ESZTENDŐT KÍVÁNUNK!

Újra itt vagyunk! Nem akarjuk dorgálni a Tisztelt Olvasót, elég sok dorgálást kaptunk, főként a 20. szám bevezetőjének utolsó mondataért, de hát ez az igazság.

Néhány dologról dióhéjban:

■ **Játékismertető:** Több, néhány mondatos ismeretűre is igény mutatkozott, ezt most megpróbáltuk összefonni sok-sok apró cheat-tel, mindent bele alapon, elvégre ez a szám kerül a karácsonyfa alá.

■ **Ismeretlen nyelvek:** Sok-sok dicséretet kaptunk a két programnyelv (*Micro-PROLOG*, 'C') folyamatos ismertetéséért, hiszen ezekről még szinte sehol sem jelent meg leírás, ezért folytatjuk is.

■ **128K:** A legfontosabb — 128K-s gépre vonatkozó — információkon már túl vagyunk, 3 csatornás hang — BASIC listákkal a továbbiakban nem töltjük meg az oldalakat, ennek sok értelme nincs. Amennyiben összegyűlik 1 oldalnyi, kimondottan 128K orientált információ, úgy azt közöljük.

■ **Gépi kód tanfolyam:** Ilyen címszó alatt megszű-

nik, szerepét a programozástechnika veszi át a továbbiakban.

■ **Rejtvény:** A többség nem kéri, ezért az itt található levelezési rovat a következő számtól oda telepszik. Ez utóbbit viszont annál többen hiányolták eddig.

■ **Térképlap:** Sajnos nem gazdaságos a pótlap legyártatása, most a Karácsonyra való tekintettel ezt az A/4 lapot még közreadjuk, ám a továbbiakban a térképeket ismét beszorítjuk a belső oldalakra. Ezzel is azt szeretnénk elérni, hogy a január 1-n esedékes jelentős nyomdai áremelkedéssel egyidőben mi ne emeljük a kiadvány fogyasztói árát!

Energilánk továbbra is véges, a jelenlegi feltételrendszerben sokkal többet nyújtani nem tudunk. Ennek ellenére nem érezzük, hogy az előző okfajtasunk óta lényeges változás állott volna be az eladott példányszám vonatkozásában. Nagyon elképzelhető, hogy 1990-ben ismét csökkentenünk kell a példányszámot (a jelenlegi 12.000-ről), ill. a megjelenési gyakoriságot egyaránt.

Last Ninja 1 hol vagy?

Tisztelt SpV! Szeretném megkérdezni, hogy a THE LAST NINJA (1) c. programot végülis áttiták a Spectrumra, mert sok újságban ill. katalógusban szerepelt. Példának okáért:

- a Spectrum Világ 4. számának 2. oldalán.

- a Sinclair Spectrum Játék és Program c. könyv 4. részének mellékletében.

- stb.

Úgyhogy ezért kérdezem, mert nem tudom, hogy végülis áttiták vagy nem. Nekem a II. része nagyon tetszett, és C64-en láttam az első részét is, és nagyon szeretném megszerezni. Ezen kívül kérném megküldeni a SYSTEM-3 nevű software forgalmazó cég címét, mert egyik újságban sem találtam. Fáradozásukat előre is köszönöm. Várom a választ.

Ifj. Gyóni Péter — Bp.

Át is írták, meg nem is, szóval mi sem tudunk többet, annyi kavargás volt a program Spectrumos változata körül. A Spectrum Világ 4. száma ill. az említett könyv 4. része kb. azonos időben jelent meg, ekkor még az angol sajtó nagy ovációval hirdette, hogy a program hamarosan megjelenik, a mi hitünk nekik. Ez azonban sajnos elmaradt, pontosabban megdőbbsen tapasztaltuk, hogy előbb utóbb végül is a 2. rész látott napvilágot, annak ellenére, hogy az angol sajtó sok-sok screen-t is bemutatott az 1. rész Spectrum verziójáról. Nos, madarak azt csiripelték, hogy a programot hazai programozók fejlesztették, a valami új programozási módszert akartak kidolgozni ebben a programban. Ezek szerint a fába bétört a fejsze...

A SYSTEM 3 címét mi sem találtuk, helyette itt a telefonszámuk: Anglia - 01-866-5692

SpV

Egy a sok közül

Tisztelt SpV! Örültem, hogy megküldték nekem az SpV 19. számát. No de nem ezért írok most önöknek.

Az Airborne Ranger-ről van szó. Miután ledobtam a három utánpótlás-csomagot, megjelenik egy nyíl és game over. Elolvastam a 19. rész ismertetését, de sajnos nem jutottam vele semmire. Jó lenne, ha néhány sorban megírnák nekem, hogy mit kell csinálni, mert ideges vagyok amiatt, hogy felvettem egy olyan programot, ami alig fért fel egy oldalra, és nem tudok vele mit kezdeni. Más. Lehet, hogy tudok küldeni egy pokedot a Cybernoid II-höz.

Gombos Bertalan,
Bonyhád

Az Airborne Ranger annál bonyolultabb játék, hogy néhány szóban bármilyen ötletet is tudnánk hozzá ajánlani. Ez a levél egy a sok közül, amelyben Olvasóink hasonló problémákkal keresnek fel bennünket a mai napig. A 17. szám borítóján már egyszer tisztáztuk, hogy inkább a Spectrum Világ-ot írjuk, mintsem, hogy napi 10-12 órában ilyen témájú levelekre válaszoljunk, mert akkor soha nem jelenne meg a Spectrum Világ. Az Airborne Ranger egyszer úgy is megérdemli, hogy leírás közöljünk róla, főleg azért, mert valóban elfoglalja egy 60 perces kazetta egy oldalát. A Cybernoid II-höz nekünk is van POKE-unk: POKE 36687,0 (sérthetetlenség), de az elmúlt 20. számban CHEAT-et is közölünk hozzá!

SpV.

SAM-betűnő kérés

Tisztelt SpV szerkesztőség! Valamelyik SpV. számban olvastam a legújabb ZX-Spectrum gépről, a ZX Spectrum SAM-ról. Érdeklődni szeret-

nék, hol, és hogyan lehet beszerezni ezt a gépet. Előre is köszönöm.

Németh Ádám, Gyál

Tisztázzuk, a SAM nem ZX Spectrum altípus, egy teljesen egyedi számítógép, amely a ZX-Spectrum ROM-programjának külön betöltésével hajlandó ZX-Spectrum-ként is működni. A gépet tudomásunk szerint továbbra sem lehet üzletben megvásárolni, csak levél útján van lehetőség megrendelni, közvetlenül, a MILES GORDON TECHNOLOGY címén keresztül. Ezt a címet már közöltük a SpV 18. számának borítóján.

SpV.

Elégedetlen gyerekek

Szerkesztőség! Egy kérésünk lenne magukhoz! Szeretnénk, hogy közöljék a legközelebbi SpV-ban a Last Ninja 2-örökletét! Köszönjük! És az az irányítás? semmi joystick, vagy talán valami kézreesső gombok. Egy ilyen programnál! Spectrumos gyerekek!

Diós Sándor, Budapest

A Last Ninja 2-nek nem öröklete, hanem örökletei vannak, vagyis mind a 6 szintnek más címen kell beállítani az örökletét: Level 1: 36578,0 - Level 2: 35993,0 - Level 3: 36571,0 - Level 4: 36514,0 - Level 5: 36393,0 - Level 6: 36822,0.

Az irányítást pedig tessék elfogadni, az van!

SpV.

Hol van az a hang...

Csösz Tibor, Veszprém

A 128K gépek tulajdonosainak általános problémája, hogy RF összeköttetésen keresztül a hang és a kép nem esik egybe a televízió készüléken. Ez főként azoknál a gépeknél

fordul elő, amelyet valaki közvetlenül Angliából hozott meg, vagy Ausztriában, ill. NSZK-ban olyan kereskedőtől vásárolta, aki közvetlenül a sziget-orozágból rendelte az árut. A probléma oka az, hogy amíg nálunk az 5,5 MHz-es differencia elfogadott a kép és hang között, addig ez a sziget-orozágból 6,5 MHz, innen adódik az eltérés. Természetesen az UHF modulátor megfelelő beavatkozással átalakítható, de az is megoldás, ha Londonban vásárolunk a géphez televíziókészüléket. Ez utóbbit esetben rádióként használhatjuk itthon a TV-t, vagy némafilmeket nézhetünk...

SpV

Embléma javaslat

Puiz András, Budapest

Annál idő után most már mi is szeretnénk volna megtalálni igazi 'hangunkat', ezt célozta a COMMODORE VILAG-gal egyidejűleg beindított új fejléc is, amelyet most már szeretnénk véglegesnek tekinteni. Akik nem ismerték volna fel, az 'S' betű a SINCLAIR-fejlécből lett átvéve. A beindított — saját tervezésű — 'S' betű szép, ezért a következő számban megmutatjuk az Olvasóknak is. Ettől függetlenül — első sorban a nyomdai munkák egyszerűsítése érdekében — a jelenlegi változat mellett maradunk.

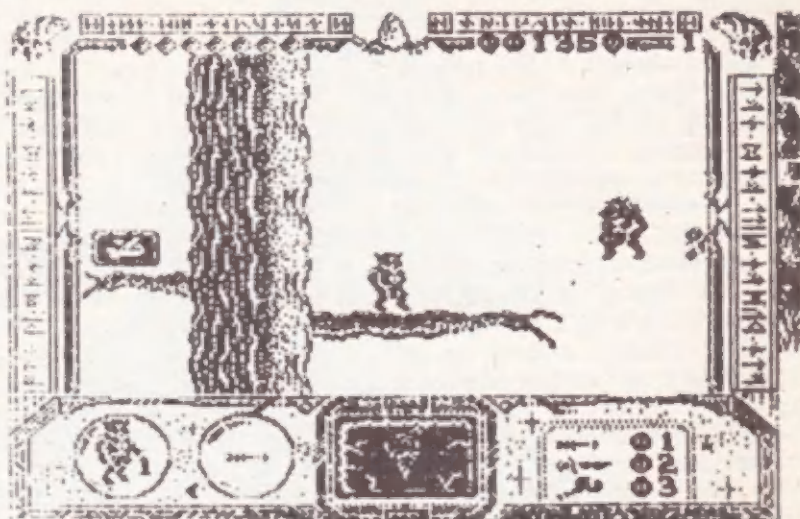
D.O.C.

Boros Péter, Győr

Az ötlek szuper, mi támogatjuk. A SYSTEM 3 telefonszámát már leírtuk. Az OCEAN címe: OCEAN Software Limited, 6 Central Street, Manchester, M2 5NS, England, az US GOLD címe pedig: US Gold Ltd., Units 2/3 Holford Way, Holford, Birmingham, B6 7AX, England. A fejleményekről szeretnénk még hallani.

SOLDIER OF FORTUNE • Firebird

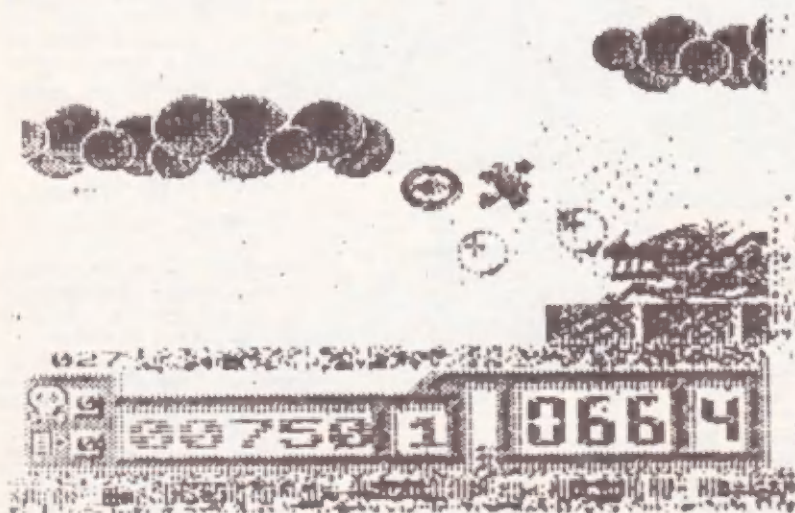
A történet főhőse, SARNAK, a Szerencse Katonája előtt nagy feladat áll: ki kell űznie az Orchard Világát előzőlő gonosz teremtményeket, le kell győznie a legfőbb Őrt. Ez csak akkor sikerülhet, ha bejárva az egész területet megtalálja a varázspirula 6 darabját, melynek segítségével összeállítható a győzelemhez szükséges anyag. Első pillantásra úgy tűnik, a játék csupán olyan elemeket tartalmaz, amelyeket már számos más játékban láttunk: mozgó és leomló járdák, teleportáló készülékek, fák, kötelek, járatok, különböző szörnyek stb. Teendők is ismert fázisokból áll: futás, harc, tárgyak gyűjtése. A szerzők (a GRAFTGOLD csoport) azonban most is – mint az URIDIUM esetében – ismert



játéktípust valósítottak meg, igen magas szinten. A fenti elemek nagy képzelőerővel és változatossággal, nagyszerű grafikával történő összeállítása emeli ki ezt a játékot az „egy a sok közül” kategóriából. A pályán számos választási lehetőségünk van, más és más úton érhetünk el ugyanarra a helyre. Az elrejtett tárgyakat mindig máshol találjuk meg, az ellenséges szörnyek egyre veszélyesebbek. Ellenük különböző erejű fegyvereket használhatunk – ha felleljük ezeket. No és persze érdemes az életadó kristályokat is begyűjteni.

Az akció-kaland játékok és a térképkészítés kedvelőinek bizonyosan jó szórakozást nyújt a Szerencse Katonája. A MULTIFACE tulajdonosok az örökéletet egyszerűen bevihetik: POKE 23314,0: POKE 46691,0.

NETHERWORLD • Hewson



A cím (Alvilág) nem a szervezett bűnözésre, hanem arra a pokoli környezetre utal, amelyben a következő egyszerű játékot játszunk: 10 szint mindegyikén előírt számú gyémántot kell adott idő alatt begyűjtenünk, kis korong alakú űrhajónkkal, majd egy teleportálón keresztül továbbjutunk a következő szintre. Mint látható, az alapötlet a BOULDER DASH-hez hasonlít.

Természetesen a fentiek végrehajtása távolról sem ilyen könnyű. Utunk során démonokkal találkozunk, amelyek ártalmas gömböcskéket szórnak. Persze ezeket lelőhetjük, és ez tanácsos, már csak azért is, mert az eltalált gömbök BONUS-szá változnak. Ezek közül némelyik hasznos

(életet, pontokat, faltörőket ad), mások kevésbé (energia és irányíthatóság elvesztése). Meg kell küzdenünk a mindenütt jelenlévő, gyakorlatilag elpusztíthatatlan mozgó aknákkal is. Problémát jelent a gyémántok megszerzése, hiszen ezek egyáltalán nem a pálya „nyilvánvaló” részein helyezkednek el: némelyik látszólag zárt területen, mások aknákkal védett szűk zugokban. Végül pedig a rendelkezésre álló idő hihetetlenül szoros, még akkor is, ha módunk van az óraüveg begyűjtésével 30 másodperccel visszaállítani az órát.

A NETHERWORLD jól játszható, szép grafikájú, simán scroll-ozó, gyors játék. Talán nem a HEWSON legjobb játéka, de mindenképpen megfelel a cég hírnevének.

REPTON MANIA • Alligata

A játék egyes szintjeihez szükséges password-ok:

| | | |
|---------------|----------------|----------------|
| A - nincs | E - SEASNAKE | I - ANNELID |
| B - ASP | F - ANEMONE | J - LEVIATHAN |
| C - CROCODILE | G - BASILISK | K - OPHIDIAN |
| D - EARTHWORM | H - CEPHALOPAD | L - KING COBRA |

FERNANDEZ MUST DIE • Image Works

Halál Fernandez-re – a játék címében szereplő diktátornak azért kell meghalnia, mert csapatai élén elfoglalta országunkat. Rajtunk a sor, hogy magányos harcosunkkal legyőzzük az idegen sőpredéket, megszabadítsuk börtönbe vetett honfitársainkat, és leromboljuk az ellenség nyolc bázisát.

A játék **COMMANDO** típusú, de annál több stratégiai elemet tartalmaz. A függőleges scroll kissé lassú, csak akkor gyorsul fel, ha sikerül gépkocsit találnunk és beszállunk. Az ütközések észlelése néha pontatlan: ellenséges golyók, teherautók haladnak át rajtunk anélkül, hogy meghalnánk. A háttér alakzatai (fák, barakkok, homokbuckák, vasutak, hidak) a szokásos árnyékolással jelennek meg.

Az ellenség leggyakoribb képviselői a gépfegyverrel folyamatosan tüzelő gyalogosok. Meg kell küzdenünk továbbá tankokkal, repülőgépekkel, és nem szabad megfeledkeznünk a számos rejtett aknáról sem. Ehhez végtelen sok gépfegyvertöltény és véges sok, de pótolható gránát áll rendelkezésünkre. Az utóbbitakkal robbanthatjuk fel például a bebörtönzött foglyok celláinak ajtaját. Lehetőségünk van egy vázlatos térkép segítségével nagyjából behatárolni pillanatnyi pozíciónkat. Végül kellemes tulajdonsága a játéknak, hogy életeink elvesztése után nem kell az egészet újra előről kezdenünk, folytathatjuk ott, ahol befejeztük.



MIND TRAP • Mastertronic

A **MIND TRAP** egy logikai játék. A cél az, hogy egy táblán levő színes kockákat sorrendbe rakjuk. Egy kerettel 4 kockát tudunk bekeretezni és a keret mozgatásával tudjuk a kockákat mozgatni illetve jobbra-balra forgatni. A keretet csak olyan irányba mozgathatjuk, amerre köröcskék vannak a képernyőn.

Mindannak ellenére, hogy az ötlet egyszerű, a játék tökéletes, és könnyen megfertőzheti az embert. A **CRASH** szerkesztői szerint ez a program jobb mint a Tetris. Míg a Tetris tetris indexe 82%, addig a **MIND TRAP**-é 84%. A programot egyébként egy jugoszláv programozó készítette (Predrag Beciric).

TIME SCANNER • Electric Dreams

Tominak hívnak? Süket vagy? Vak vagy? Röviden érzed-e úgy magad mint a "pinball wizard"? Ha minden kérdésre pozitív a válaszod, akkor ez a program, a **Time Scanner** a Te játékod. Ez a flipper valóban fantasztikus. Egyszerre 6 labdával is játszhatunk és be van építve a "TILT" is. Minden tábla két képből áll, amely nehezíti a játékot.

Everyone's A Wally • Mikro-Gen

Akik **MULTIFACE**-szel rendelkeznek az örökélet **POKE**-ot könnyen bevihetik:

POKE 58217,0: POKE 58218,0: POKE 58219,0

Az értékeket közvetlenül a játék betöltődése után kell beírni.

A bevittelt közöljük azok számára is, akik ■ 319/209/32768/16165 file-térképes verzióval rendelkeznek.

MERGE™ segítségével töltjük be a **BASIC** loader-t, majd módosítsuk a következők szerint:

50 LOAD"" CODE

51 POKE 65345,88

52 POKE 65346,255

53 FOR f=65368 TO 65376

54 READ a: POKE f,a: NEXT f

55 FOR f=65400 TO 65415

56 READ a: POKE f,a: NEXT f

60 RANDOMIZE USR 55000

72 DATA 33,120,255,34,34,255,195,12,255

73 DATA 205,128,91,175,50,106,227,50,106,227,50,107,227,195,132,129

RUN, majd indítsuk ■ magnetofont tovább. Betöltés után végtelen élettel rendelkezünk, bármelyik szereplővel is játszunk.

WULFAN • Mastertronic

Ez egy gondolkodtató játék. A játék végére elég nehéz eljutni, még egy térkép segítségével is. Cél: hogy megtaláljuk a "rosszalkodót" és átadjuk neki bukósisakot. A labirintus tele van ajtóval, amit kulcsokkal kell kinyitnunk.

A játékban a következő tárgyak szerepelnek:

- bomba: néhány falat ledönt
- buzogány: váltásra szolgál
- üveg: ezzel lehet lefizetni Mogdun embereit, akik általában cserébe kulcsot adnak.
- pénz: váltásra szolgál
- kulcs: többféle létezik: vannak amelyek csak egy bizonyos ajtót nyitnak, vannak olyanok is, amelyek több nyitására szolgálnak.

GILBERT-ESCAPE FROM DRILL • Again-Again

Gilbert egy földönkívüli, idegen, szimpatikus zöld hős a *Get Fresh* és a *Gilbert's Fridge* játékokból. Az új játékban megbízást kapott valahol a kozmoszban. A sajátjai (*Drillansok*) szétszedték űrhajóját és szétszórták. Főhősünknek 24 óra áll rendelkezésére, hogy összeszedje azt darabjaiból. A játék a *Pyjamarama*-hoz hasonló, jó grafikával és animációval.

Sir Fred • Mikro-Gen

A "*Pofonok völgye*". Így jellemezhetnénk tömören azt, amit levélíróinktól ■■ elmúlt időszakban sorozatosan kaptunk, a már legutóbb is "*Ierágott csont*"-nak nevezett **SIR FRED** örökélet bevételével kapcsolatban. Nos engedje meg ■ Tisztelt Olvasó, hogy most mindent jóvátegyünk, vagyis mindhárom verzióra ismertetjük a **HELYES** beviteli módszert!

1. MULTIFACE törés

A SpV. 17. számának 9. oldalán található beviteli módszert szeretnénk kiegészíteni: A POKE 46650,183: III. a RANDOMIZE USR 24833 (ENTER) utasítások közé szúrjuk ■■ a következőket: ... POKE 24012,194: POKE 24013,6: POKE 24014,93: POKE 24015,0: POKE 24016,0: POKE 24017,0: POKE 24018,0: ...

2. BASIC/48983 file térképes verzió

A SpV. 6. számának 2. oldalán közölt beviteli módszer esetén ■ következő változtatásokra van szükség:

20 FOR I=65400 TO 65474. READ a: POKE I,a: NEXT I

50 DATA 5,210,0,0,245,62,183,50,58,182

60 DATA 62,194,50,204,93,62,6,50,205,93

70 DATA 62,93,50,206,93,175,50,207,93

80 DATA 50,208,93,50,209,93,50,210,93

■■ DATA 241,195,68,181

100 RANDOMIZE USR 65400

3. FUTURESOFTE verzió

A SpV. 18. számának 9. oldalán található módszer BASIC listája helyesen:

10 CLEAR 65399

20 FOR I=65400 TO 65449

30 READ a: POKE I,a: NEXT I

40 DATA 49,253,255,221,33,0,91,17,0,250,62,0,55,205,86,5

50 DATA 62,183,50,58,182,62,194,50,204,93,62,6,50,205,93,62,93,50,206,93,175,50,207,93,50,208,93,50,209,93,50,210,93,201

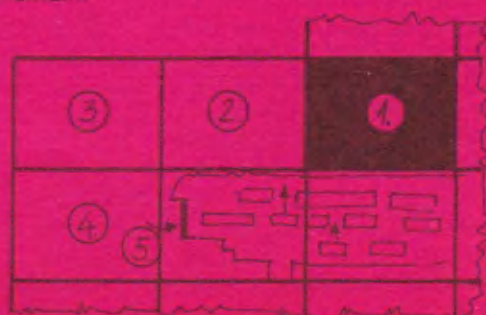
60 POKE 65533,68: POKE 65534,181

70 STOP

Ezen módosítások után nemcsak FRED-ünk lesz örökifjú, hanem a **SPACE** billentyű megnyomására ■ Játék újraindul. Ez bizony kellemes dolog, mert leesni a kút mögé gyufa nélkül...

Egy kis csalás: Sokan rájöhettek, hogy a 9 nyílvezető egy kicsit kevés, azonban ha a lövést úgy adjuk le, hogy nem a gombot engedjük el, hanem a megfelelő szögnél egyszerűen átváltunk egy másik tárgyra, akkor el is röpül a nyílvezető és meg is marad! Ez az ötlet ■ kőnél sajnos nem jött össze.

A 6 számban közölt térkép is hiányos volt egy helyen, ugyanis ■ sötét szobától (1) ■ varázslón (2), erkélyen (3) és a királynő háloaszobáján (4) át egy szekrényen keresztül (5) titkos átjárót találunk!



R-Type - Electric Dreams

Az **R-Type** az egyik legszebb, és ugyanakkor legnehezebb arcade játék. Éppen ezért úgy gondoltuk, hogy érdemes a játékot megkönnyíteni térkép, néhány útmutatás és **MULTIFACE POKE** segítségével.



- 1.szint: A nagy gyűrűt kék pontjának eltávolításával tehetjük ártalmatlanná, esetleg úgy, hogy lekapcsoljuk (DETACH) a szondát és előre küldjük a gyűrű belsejébe. A szint végén először a szörny négy szemét lövük ki, majd az előbukkanó fejet sorozzuk meg.
- 2.szint: Óvakodjunk a nagy kígyótól. A szív a tetején megjelenő kék gömb sokszori meglovésével robbantható fel, de gyorsabb, ha az űrhajónk orrára kapcsolt szondával ráereszkedve megérintjük.
- 3.szint: A hatalmas ellenséges űrhajó tetején ki-be mozgó alkatrészt kell többször eltalálnunk, vagy megérintenünk.
- 4.szint: Irsuk az állandóan növekvő zöld falat. A szint végén megjelenő űrhajó 3 darabra válik, ekkor az egyes részek zöld pontjait célozzuk meg.
- 5.szint: A rejtekéből végül előbukkanó űrhajó a középpontjára a legérzékenyebb.
- 6.szint: A nagy mozgó tömböket a középpontjukon, vagy hátulról kell többször eltalálnunk. A szint végén „csak” ki kell tartanunk.
- 7.szint: A szint végén a jobb oldalon megjelenő alakot kell bombáznunk, de az is lehetséges, hogy itt is elég bizonyos ideig élve maradnunk. Pajzsok nélkül nincs esélyünk.
- 8.szint: Ugyanaz, mint a 7.szint, de nehezebb.
- 9.szint: Aki idáig eljutott, annak már nincs szüksége útmutatásra.

Néhány hasznos, és érdekes POKE:

| | |
|-----------|---|
| 37374,0 | – végtelen sok élet (bár ez önmagában nem sok segítséget jelent) |
| 37362,201 | – sérthetatlenség (az űrhajónak semmi sem árt) |
| 38240,0 | |
| 38241,0 | |
| 38242,0 | – űrhajónk eltűnik, így sérthetatlenné válik az ellenséges lények számára (a háttérbe való ütközés azonban most is halálos) |
| 38241,6 | |
| 38242,154 | – megfordítja az űrhajót vízszintes tengelye körül |
| 38241,14 | |
| 38242,154 | – megfordítja az űrhajót függőleges tengelye körül |
| 38241,22 | |
| 38242,154 | – megfordítja az űrhajót mindkét tengelye körül |
| 38241,254 | |
| 38242,153 | – visszaállítja az eredeti helyzetet |
| 34930,195 | – eltünteti a háttérrel |

Végül a következő néhány byte bevitele igen hasznos kiegészítést jelent. A 'W' billentyű lenyomására az űrhajó teljes fegyverzettel rendelkezik:

| CÍM | KÓD |
|------|-------------------------|
| 5B00 | 01 FE FB ED 78 E6 02 20 |
| 5B08 | 0B 01 1D 00 11 A3 7A 21 |
| 5B10 | 18 5B ED B0 C3 79 89 00 |
| 5B18 | 01 03 01 01 01 00 0B 08 |
| 5B20 | 00 0C 0A 01 04 09 02 06 |
| 5B28 | 02 0D 0A 07 06 0A 0E 00 |
| 5B30 | 00 00 03 03 00 00 00 00 |

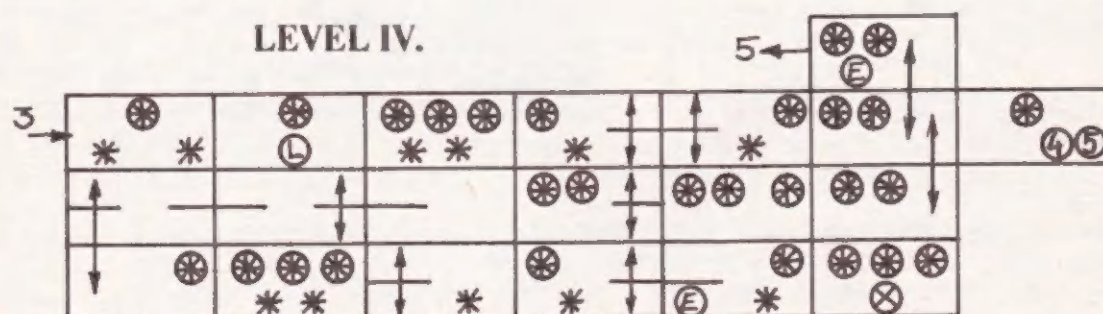
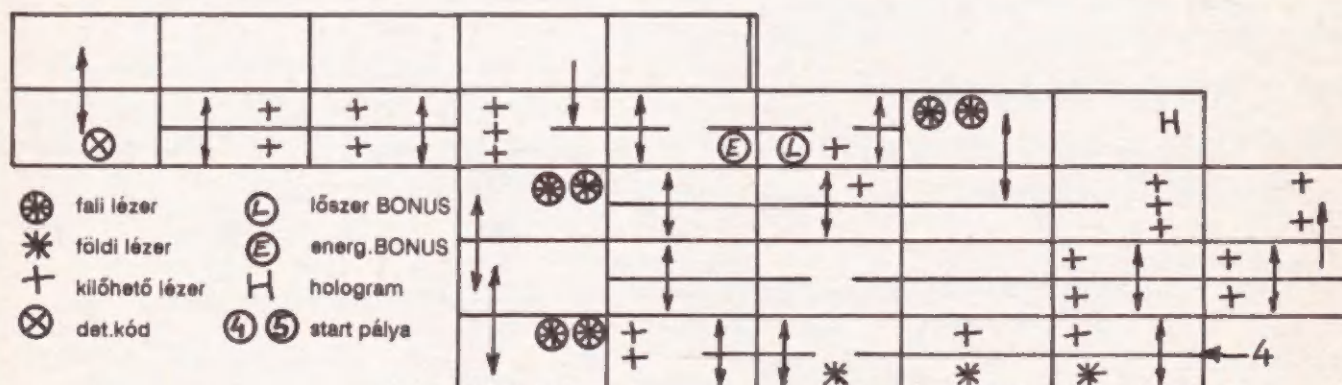
Ezután a következő két POKE-ot még vigyük be: 34388,0: 34389,91

BLASTEROIDS • Mirrorsoft

Valahol a világűrben bolyong egy magányos űrhajó anyabolygója után kutatva. Útközben mindenféle ellenfelek akadályozzák. Ez a program egy tipikus lövöldözős játék.

TITAN • Titus

Ez a program két ismert játék a *THROUGH THE WALL* és a *GAUNTLET* keveréke. A játék egy labirintusban játszódik ahol téglák után kutatunk. A cél az, hogy egy labdával a téglákat leszedjük. A játék közben nem kell ügyelnünk arra, hogy ha a labda elmegy mellettünk, akkor elveszítjük életünket. Amikor rátalálunk egy téglára, akkor vezessük oda a kuglit és üssük le. De ez még nem minden. Van olyan téglá is, amit nem a labdával lehet elintézni. Ezenkívül különböző zavaró ellenfelek is vannak. Vannak olyan téglák, amelyeket nehéz eltalálni, mert mozognak, a halálfejjel megjelöltek pedig veszélyesek is számunkra. Vannak kicsi és nagy felületű labirintusok. A játéknak 70-80 szintje van.

DAN DARE • Virgin (Gang of Five)**LEVEL V.****BEDLAM • GO!**

Amikor meghalunk, nyomjuk meg ■ <C> billentyűt, ekkor visszakérülünk ■ "élő" pozíciónkba, sőt életeink száma is maximális lesz.

AQUASQUAD • Atlantis

Amikor az üzenetek scroll-ozódnak ■ képernyőn, nyomjuk meg ■ <SYMBOL SHIFT> és ■ <C> billentyűket egyszerre, majd gépeljük be: 726549, s lám sérthetetlenek és örökifjúak leszünk a játékban.

KOSMOS • Atlantis

Amikor játszunk, nyomjuk meg a <kurzor le> billentyűt, s megjelenik ■ menü. Most nyomjuk meg ■ <SYMBOL SHIFT> és a <K> billentyűket egyszerre az örökélethez.

SUPER KID • Atlantis

A főképernyőn nyomjuk meg a <G>, <D> és az <F> billentyűket, ekkor jutalomként örökéletünk lesz.

CAPTAIN FIZZ • Psyclapse

Stratégiai játék, hasonlóan a *Laser Squad*-hez. Két játékos játszhatja. A cél az, hogy az Ikarus nevű bolygóra kitelepítsünk egy fajt, az un. *Blaster-Tront*. Fontos, hogy a két játékos egyesítse erőit, mert csak így juthatunk el a játék végére.

CRAZY CARS 2 • Titus

Éppen hogy csak kipihentük magunkat az első résztől, amikor a TITUS programozók meghírdették a második részt. A programkészítők szerint ez a program jobb mint az előző (fékezésnél, robbanásnál, 360%-os fordulásnál). A Ferrari legújabb típusú autóját, az F40-eset hajtjuk. Néha rendőrautó zavarja meg kocsinkat, lgyekeznünk kell, hogy ne érjen utol bennünket. Az úton különböző zavaró tárgyak vannak. Annak a valószínűsége, hogy egyiknek se menjünk neki 1:1000000000000. Sok sikert...

ORIENTAL GAMES • Firebird

A *Firebird* megint valami újat készített nekünk, de az ötlet régi. A távolkeleti játékokban négyféle versengési csoport van: Kung Fu, Sumo birkózás, Kendo és az un. hollywoodi játékok. Ha mind a négyből győztesen kerülünk ki, akkor eljutunk a az Activision legújabb GRAND versenyre, ahol elnyerhetjük a *Grand Master*-i címet.

TRANTOR THE LAST STORMTROOPER • GO!

A játékban használt kódok ■ következők: KEMPSTON, JOYSTICK, SPECTRUM, SOFTWARE, KEYBOARD, COMPUTER, CASSETTE, SINCLAIR, GRAPHICS, HARDWARE, TERMINAL, PRINTERS, CONTROLS, WARGAMES, WARRIORS, MEGAGAME

CHICAGO 30S • US Gold

Amikor a játék betöltődött, nyomjuk meg a PAUSE gombot (<H>), majd <1> – <9>-ig a kívánt szintnek megfelelő számbillentyűt, s a kiválasztott szinten folytathatjuk ■ játékot.

GEMINI WING • Virgin

A szintek közötti password-ok ■ következők:

| | | | | | | | |
|----------|-----------|----------|----------|----------|----------|----------|----------|
| Level 1: | THE START | Level 3: | WHATWALL | Level 5: | SKULLDUG | Level 7: | CREEPISH |
| Level 2: | EYEPLANT | Level 4: | GOODNITE | Level 6: | BIGMOUTH | Level 8: | FINALFXS |

LAST MISSION • US Gold

A végtelen űrhajóhoz válasszuk ki az 1 játékos játékot, majd nyomjuk meg az " (idézőjel) billentyűt.

RED HEAT • Ocean

Nyomjuk meg ■ <SYMBOL SHIFT> billentyűt, valamint az összes számbillentyűt egyszerre. 10 élet és még egy-két meglepetés lesz az eredménye!

HUMAN KILLING MACHINE • Capcom

Ha nem tetszik a képernyő színe, akkor nyomjuk meg a <C> billentyűt, és változtassuk meg azt!

SABOTAGE • Zeppelin

A Password kódok a következők:

| | | | |
|----------|-------------|----------|--------------|
| Level 1: | nem kell | Level 5: | ONOMASTICS5 |
| Level 2: | BUMBLE BEE2 | Level 6: | SALMAGUNDI6 |
| Level 3: | HONORARIUM3 | Level 7: | PSEUDONYMOUS |
| Level 4: | PHENOMENON4 | Level 8: | ONOMATOPOEIA |

THE 'A' TEAM • Zafiro

A játék 2. szintjének kódja: WAEONDPEAER

MEGANOVA • Dinamic

A játék 2. szintjének kódja: 26719, a 3. szint kódja: 16640

NAVY MOVES • Dinamic

A játék 2. szintjének kódja: 63723

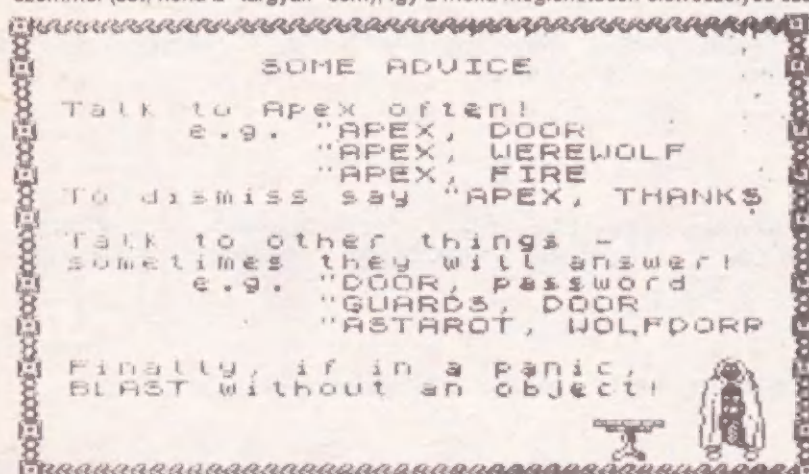
SOL NEGRO • Opera Soft

A játék 2. szintjének kódja: 2414520

A **MARSPORT** megoldásának ismertetésekor (SpV. 14. szám) már ijesztgettük Olvasóinkat ama szörnyűséggel, hogy alkalomadtán sort kerítünk a **GARGOYLE GAMES** ezidáig utolsó arcade/adventure játéka is. Ez a fel-elő (?) pillanat most érkezett el: mindenki fedezékbe! — itt jön a **HEAVY ON THE MAGICK**. Ez a játékprogram már mindennek a teteje: kb. olyan nehéz, mint a **TIR NA NOG/DUN DARACH/MARSPORT**-trilógia együttvéve, pedig azok külön-külön is elég sok kellemetlenséget okozhatnak minden vállalkozónak.

A játék kivitelezésében az említett **GARGOYLE**-trilógiához képest némi változás tapasztalható: a hagyományos adventure-típusú játékokhoz hasonlóan az irányítás a képernyő egy elkülönített részén szövegbegépeléses módszerrel történik. A játék egyébként iskolapéldája lehetne az úgynevezett "interaktív" kezelésű kalandjátékoknak, mert az általunk irányított figurának kiadott utasításaink azonnal végrehajthatók, a hatás a képernyőn is rögtön le is mérhető. A szöveges újításból kifolyólag a **HEAVY ON THE MAGICK**-ben fokozottabban van szükség az angol nyelv ismeretére (no meg persze egy jó adag műveltségre, intelligenciára és asszociációs mámorra). Mielőtt a Shakespeare nyelvében kevésbé járatos olvasóinkat végképp elriasztanánk a játéktól, közölünk, hogy egy angol-magyar szótár megoldja a problémákat, sőt a szöveges kommunikáció is meglehetősen kényelmes módszer alapján zajlik. Talán további bevezetőre nincs is szükség, hiszen a játékot mindenki ismerheti, lévén megjelenésének dátuma **Sinclair után 4.** (műveletleneknek és C-64 tulajdonosoknak: 1986.)

Főhősünk az **Axil the Able** névre hallgat, foglalkozását tekintve a Búbaj Búbaj Kft. ügyvezető igazgatója. Vezetéknévéről (the Able) ítélve valamilyen feladatra rátermett férfiú — bár a feladat mibenlétét egyelőre sűrű homály fedi. Az viszont egészen bizonyos, hogy pillanatnyilag éppen egy 256 helyszínből álló, elvarázsolt barlangrendszerben rontja a levegőt, és mindenféle őrdögi praktikákat követ el a barlangban tartózkodó élőlényekkel és tárgyakkal (mikor mi akad az útjába). Természetesen az élőlények ezt általában nem nézik jó szemmel (sőt, néha a "tárgyak" sem), így a móka meglehetősen életveszélyes számára.

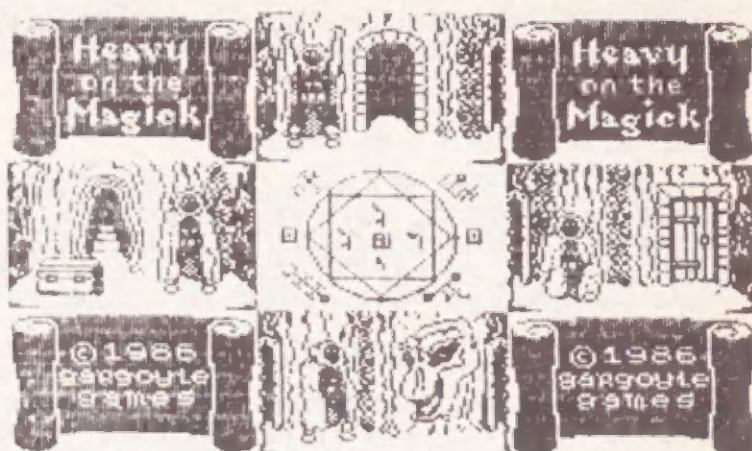


hogy játszott **GARGOYLE**-játékkal, annak nem fog sok meglepetést tartalmazni, bár néhány újdonság ebben is található:

- A **MAGICK** választásával indíthatjuk/folytathatjuk a játékot a pillanatnyi állásból.
- A **SAVE GAME** választásával kimenthetjük a játék pillanatnyi állását. Szokás szerint egy betűjelet kell megadnunk névnek. A kimentett állás a **RESTORE GAME** opcióval olvasható vissza, itt a töltési kívánt állást jelző betűt kell megadnunk. Ha a töltés sikertelen, vagy közben megnyomjuk a **"SPACE"**-t, a program **ABANDONED** (feladva) felirattal visszaáll a főmenübe.
- A **SAVE/RESTORE AXIL** opciókkal a szereplőnk pillanatnyi tulajdonságait menthetjük ki/tölthetjük vissza. Mivel alapvetően ezek a tulajdonságok határozzák meg, hogy milyen sikerrel ténykedünk a játékban, az egy nagyon hasznos opciópár: lehetőségünk van ugyanis arra, hogy a játékokat módosított (azaz jobb) tulajdonságokkal kezdjük el. Ha tulajdonságokat töltnék be, a játék mindig előről kezdődik! — tehát nincs lehetőség arra, hogy egy bizonyos helyen sok energiát veszítve "feltuningoljuk" Axil-t. A pillanatnyi tulajdonságok (**STAMINA/SKILL/LUCK**), tapasztalati pontok (**EXPERIENCE POINTS**) és varázslói besorolás (**GRADE**) a menü jobb oldalán láthatóak — jelentésükkel a későbbiekben még részletesen foglalkozunk.
- A **REALIGN STATUS** használatával felcserélhetjük egymással a tulajdonságok értékeit.

Az utóbbi három opció megfelelő keverésével elég jól felkészíthetjük Axil mestert az útra. Felhívánk a figyelmet a játék egy érdekességére: ha Axil meghal (elfogy az összes életereje), akkor a **MAGICK** választásával — bár a játék a startszobából indul újra és az összes tárgy is visszakerül a helyére — nem előről kezdődik a játék, mert tapasztalati pontjaink, illetve az esetlegesen elért magasabb varázslói besorolás megmarad. A játék egyébként figyelembe veszi ezt a paramétert a kezdeti tulajdonságok beállításánál is. Most nézzük sorba milyen jellemzői vannak Axil barátunknak:

- STAMINA:** ez az életerőnk, ha elfogy, Axil meghal. Az életerő pontok a parancsok kiadásával (tárgy felvétele, mozgás, stb.) párhuzamosan csökken. Ugyanezt csökkentik az ellenfelek támadásai is, méghozzá annál nagyobb mértékben, minél nagyobb a támadási jellemzőjük. Egyes akadályokhoz, szellemekhez vagy ellenségekkel való érintkezés közben szintén nagyon gyorsan csökken. Növelhető viszont néhány, ételként funkcionáló "tárgy" (pl. a kenyér) felvételével illetve transzfúzió (ld. később) segítségével.
- SKILL:** Axil ügyességének mértékét jelzi. Növelhető a későbbiekben használható tárgyak (pl. a Grimoire-varázskönyv) felvételével, csökkenni csak akkor szokott, ha elvesztjük az életerőnket és új játékot kell kezdenünk.
- LUCK:** Ez a szerencsénket jelzi, legfőképpen arra van hatással, hogy a támadásaink mennyivel csökkentik az ellenfelek életerejét, illetve azok milyen sűrűséggel támadnak vissza. Növelhető mindenféle kabalaként használható tárgy felvételével, de minden új játék kezdetekor — az előbbihez képest — 1-el csökken (ha 1 volt, akkor kisebb STAMINA-val vagy SKILL-el indulunk).



Betöltődés után egy kis útmutatót kapunk a játékhoz a programozóktól: eszerint gyakran kell majd beszélünk Apex-szel (kommunikációs útmutató mellékelve), valamint nem árt, ha egyéb dolgokkal is csevegést kezdeményezünk, mert néha "választ" kapunk tőlük. A "választ" természetesen nem mindig információ, hanem néha cselekmény formájában jelentkezik. Ezután egyébként három példa is található, amely egyébként kiváló tippként szolgál a játékban azoknak, akik egyedül próbálják meg végigjátszani a játékot: innen lehet megtudni a zárt ajtók kinyitását és a szellemek megidézésének módját. Az útmutatás végén egy elmés megállapítás található, mely szerint ha megijedünk, akkor csak robbantsunk tárgy nélkül. Örömmel...

Egy billentyű megnyomása után bejelentkezik a játék főmenüje, ahol az egyes pontok előtt álló számnak megfelelő számbillentyűvel változtathatunk a opciók között. Aki már elkövette az a hibát,

EXPERIENCE POINTS: Tapasztalati pontok. Az elnevezés magáért beszél, az addigi játék során összegyűjtött tapasztalatoknak a kijelzése. Ezt általában más előlányek kiiktásával, egyes tárgyak felvételével, illetve valamilyen más cselekménnyel növelhetjük. Ha valamilyen cselekedetünk plusz tapasztalati pontot eredményez, akkor a növekedést a program a jobb alsó sarokban lévő ablakban jelzi. A tapasztalati pontok a létező elfogyása után is megmaradnak, és a későbbiekben – transzfúzió útján (ld.később) – létező pontokká is átalakíthatjuk őket.

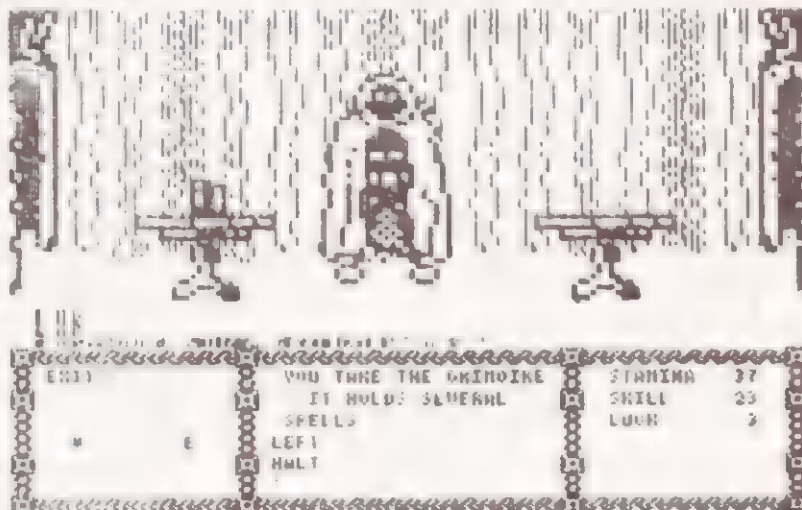
GRADE: varázslói besorolás. Ehhez a játékban egy viszonyítási szám is járul, ami talán azt jelzi, hogy Axil-nak milyen esélyei vannak a feladat végrehajtását illetően. Ennek megfelelően a kezdeti (azaz leggyengébb) NEOPYTHE besorolás viszonyzáma 1:10, míg a következő szintre (ZELATOR) 2:9, és így tovább. A varázslói besorolás a nehezebb feladatok megoldásával (pl. kulcsszóval nyitható ajtók jelszavának megfejtésével) növelhető, amit a program szöveggel és a képernyő villogtatásával is jelez. A tapasztalati pontokhoz hasonlóan, a rangsorolás is megmarad egy játék elvesztése után.

Az elvárszolt barlangrendszer a különálló szintből áll, amelyek mindegyike 64 helyszínt tartalmaz. Az egyes szintekre a program számokkal hivatkozik, az egyes helyszíneknek is általában saját nevük van (ezeket a térképleapon lévő térképen külön fel is tüntettük [a későbbiekben ugyanis szükség lesz rájuk], más szintre vezető kijáratokat a szint száma jelzi). Egy játék elindítása után a barlangrendszer 2 szintjén találjuk magunkat. **Nyomor Szobájában (ROOM OF MISERY)** A képernyőt négy részre oszthatjuk a felső kétharmadban látható a helyszín és az ott tartózkodó szereplők, az alsó részen pedig ablak kapott helyet, amelyek az alábbi információkat tartalmazhatják:

■ Az első ablak a 'Z' billentyűvel kérhető parancsok kijelzőjeként működik, ezekkel a továbbiakban részletesen foglalkozunk.

■ A középső ablakban a billentyűnyomással előállított vagy begépelte szövegeket, parancsokat illetve a játéknak egyes eseményekre vonatkozó kommentárjait és információit láthatjuk.

■ A jobb oldali ablakban láthatóak Axil bátyó pillanatnyi tulajdonságainak értékei (STAMINA/SKILL/LUCK), illetve ha valamilyen más élőlény tartózkodik a szobában, akkor Axil tulajdonságai alatt megjelenik a neve, életereje (STAMINA) és ügyessége (CUNNING). Ez utóbbi egyébként arra vonatkozik, hogy ha megtámad bennünket, akkor kb. mennyivel fogja csökkenteni a létezőnket.



Mint már említettük, az irányítás az egy billentyű megnyomásával elérhető standard parancsokkal és – ha szükség van – annak a tárgynévnek a begépelésével történik, amire a parancs vonatkozni fog. A kurzort egy kis kör jelöli, a parancs végrehajtása a 'RETURN' megnyomásával kezdődik el. Több részből álló parancssorozat is folyamatosan végrehajtható, ha a parancsokat elválasztva gépeljük be (pl. EAST,EAST,RIGHT,PICK UP BAG) vagy minden egyes parancs kiadása után megnyomjuk a 'RETURN'-t. A két módszert kombinálva is használhatjuk. Parancssorozat kiadása esetén Axil folyamatosan hajtja végre a parancsokat addig, amíg el nem fogynak vagy akadályba illetve ellenséges élőlénybe nem utkozik.

Ha olyan parancsot akarunk végrehajtani, ami pillanatnyilag vagy abszolút nem lehetséges (pl. nem létező kijáraton akarunk kimenni vagy olyan tárgyat akarunk felvenni, ami nincs a szobában), Axil felénk fordul és tanácstalanul széttárja a kezét.

Bármilyen parancs(sorozat) végrehajtása azonnal megszakítható a 'H' billentyű megnyomásával elérhető HALT parancs segítségével. Ilyenkor Axil azonnal megáll és vár a további parancsokra. A HALT használatára például akkor van szükség, ha észrevesszük, hogy egy helyszínen áthaladva Axil valamilyen akadályba vagy ellenségbe utkozik (azaz villámgyorsan elvesztene életerejét).

A mozgás az angol égtájak rövidítéseinek megfelelő billentyűkkel történik, 'N'(ORTH): észak, 'E'(AST): kelet, 'S'(OUTH): dél, 'W'(EST): nyugat. Dél-nyugati (SOUTH-WEST) irányt úgy állíthatunk elő, ha a 'S' billentyű után a 'W'-t is megnyomjuk. Előfordulhat, hogy egy helyszínről nem akarunk kimenni, de valamilyen ok miatt (pl. közeleg ellenség vagy egy távolabb lévő tárgy) kénytelenek vagyunk helyet változtatni: ilyenkor használható a 'R' illetve 'L' billentyűkkel elérhető RIGHT és LEFT parancs, amelyek hatására Axil szép komótosan átballag a helyszín jobb illetve oldalára. A LEFT/RIGHT és a HALT parancsok kombinált használatával pontosan egy adott helyre is állíthatjuk barátunkat.

Nézzük sorban a további parancsokat:

EXAMINE ('X' billentyű): Nagyon hasznos funkció, egy tárgyat vizsgálhatunk meg vele. A parancs után kell gépelnünk a tárgy nevét, amit meg kívánunk vizsgálni (ha ilyen tárgy nem létezik vagy a nevet rosszul adtuk meg, Axil széttárja a kezét és egy EXAMINE WHAT? kérdéssel érdeklődik azután, hogy ugyan mit kéne megvizsgálnia). Ha a tárgy nevét jól adtuk meg, Axil odaballag hozzá és a középső ablakban közli az észrevételeit róla. A tárgyakat két okból is célszerű megvizsgálni:

■ információkat tudhatunk meg róluk, pl. valamilyen szó van rájuk véve, ami a későbbiekben hasznos lehet számunkra, esetleg megtudjuk milyen anyagból vannak, stb. Néha ugyan Axil barátunk meglehetősen épületes megállapításokat tesz (pl. EXAMINE SIGN (Vizsgáld meg a jelet!) parancsra közli velünk, hogy IT'S A (Ez egy jel)), de néha olyan fontos szavakat is megtalálhatunk, ami akkor nem jutna a tudomásunkra, ha csak simán felvennénk a tárgyat.

■ néhány tárgyból több is megtalálható a játékban. Ezek közül nem mindegyik az, aminek látszik, ilyenkor a vizsgálat eredményéről Axil úgy tájékoztat bennünket, hogy "úgy néz ki, mintha ... lenne" (IT LOOKS LIKE A ...). Az ilyen tárgyak egész biztosan nem azok, aminek látszanak: egy részük meg van mérgezve (felvételükkor egy csomó STAMINA-pontot veszünk) és általában semmire sem jók. Például rögtön kezdéskor két egyforma könyvet láthatunk a két asztalon. Vizsgálatuk után a következők derülnek ki róluk: a jobb oldali a varázslók kedvenc felhasználói kézikönyve, amelynek felvételével három varázslat birtokába juthatunk és a ponttal növelhetjük a SKILL értékét, a másik viszont csak könyvnek tűnik – meg is van mérgezve. Természetesen nem lenne egy igazi GARGOYLE-játék, ha ellenkező példa nem akadna: olyan dolgoknál, amelyek alapvetően káros hatással vannak ránk, a pont fordítva van. Például arról a tárgyról, amelyik "úgy néz ki, mint egy csörgő kígyó" (IT LOOKS LIKE ROCKSNAKE), felvétele után kiderül, hogy egy vascsat (IRON CLASP), amely szerencsét hoz és növeli a tapasztalati pontokat – az igazi csörgő kígyó természetesen jól megmar bennünk, ha felvesszük. Ennyit a GARGOYLE-féle negatív logikáról, konzekvencia: mindent meg kell vizsgálni.

Ha nem tudjuk a tárgy nevét, akkor használjuk az EXAMINE OBJECT parancsot. Erre Axil a hozzá legközelebb lévő tárgyat vizsgálja meg, ami nem feltétlenül felvehető tárgy (lehet berendezési tárgy, jel a falon, vagy egyéb más). Ha nem a kívánt tárgyhöz ment oda a szerencsétlen, akkor a LEFT/RIGHT és a HALT parancsokkal állítsuk pontosan elé. Ha mást nem, legalább a nevét biztosan megtudjuk.

PICK UP ('P' billentyű): Tárgy felvételére szolgál. A tárgy nevét kell megadnunk utána, amelyet fel akarunk venni (ha a helyszínen nincs ilyen tárgy, vagy nem adtuk meg a nevét, Axil **PICK UP WHAT?** kérdéssel érdeklődik, hogy mit kívánunk felvenni). Ha a nevet jól adtuk meg, Axil odaballag a tárgyhöz és felveszi, amennyiben van az erszényben még hely. Összesen a tárgy lehet egyszerre nálunk, ha ezután is fel akarunk még venni valamit, akkor a program közli, hogy a erszény tele van (**THE POUCH IS FULL**). Nem számítanak tárgynak a felvett ételek (ezeket Axil ugyanis rögtön bekebelezi; célszerű tehát egy tárgyat letennünk, ha a erszényünk tele van, de tudjuk, hogy a közelünkben lévő tárgy biztosan étel – miután Axil elfogyasztotta a kaját, ismét felvehetjük a tárgyat) illetve a **GRIMOIRE** varázskönyvhöz tartozó két lap (**SCROLL OF PARCHMENT** – ezek beépülnek a **GRIMOIRE**-ba). Ha olyan tárgyat akarunk felvenni, amelyet nem lehet (pl. sziklát vagy oszlopot), a program közli, hogy nem tudjuk felemelni (**YOU CAN'T LIFT ...**). Egyébként ezekre nincs is szükségünk. Még egyszer felhívánk a figyelmet arra, hogy nem árt a **EXAMINE**-nal megvizsgálni azokat a tárgyakat, amelyeket fel akarunk venni! Ha a tárgy nevét nem tudjuk, itt is használhatjuk a **PICK UP OBJECT** formulát – Axil hozzá legközelebb eső tárgyat próbálja meg felvenni. A **LEFT/RIGHT** és **HALT** parancsok használatával pontosan a kívánt tárgy elé állhatunk. Megjegyzendő, hogy minden egyes tárgy felvétele egy **STAMINA**-pontba kerül.

DROP ('D' billentyű): a **PICK UP** ellentéte, egy tárgyat tehetünk le vele a földre. Ha nincs nálunk ilyen tárgy vagy adtuk meg a nevét, Axil **DROP WHAT?** kérdéssel tudakolja, hogy mire gondolunk.

OPTIONS ('O' billentyű): Használatával visszatérhetünk a játék főmenüjébe, ahonnan kimenthetünk/visszatölthetünk játékállást vagy Axil paramétereit – esetleg időt nyerhetünk a gondolkodáshoz. A játékot a **MAGICI**-el folytathatjuk tovább. Ha a helyszínen rajtunk kívül tartózkodik még egy élőlény vagy démon, illetve olyan szobában vagyunk, ahol varázslói szintugrás történt, nem lehet visszaugrani a menübe (**OPTIONS? NOT NOW!**), vagy át kell mennünk egy üres helyszínre, vagy meg kell dinünk előbb az élőlényt.

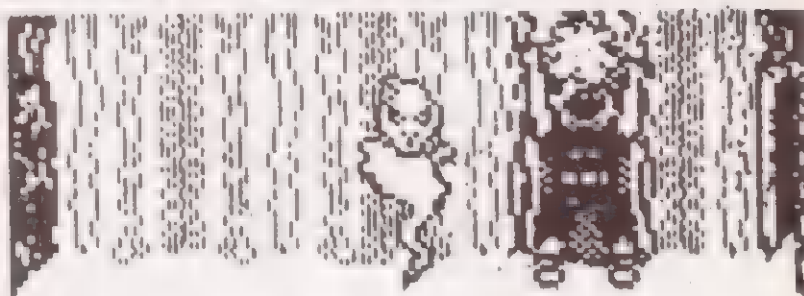
EXITS ('Z' billentyű): A 'Z' billentyű nyomogatásával négy parancsot érhetünk el, amelyek a bal oldali ablakban a helyszínre illetve Axil a vonatkozó információkat jelenítenek meg. Az **EXITS** parancsra az ablakban az aktuális helyszínről nyíló ajtók iránya jelenik meg. A program nem tesz különbséget a pillanatnyilag használható illetve használhatatlan (pl. kulcsszóval nyitható vagy egyirányú) ajtók között – mindegyiket jelzi. Néha az egyes kijáratoknál nyílak is megjelennek, amelyek utalnak, hogy a ajtón keresztül magasabb (↑) vagy alacsonyabb (↓) számozású szintre lehet átjutni. Megjegyzendő, hogy a program automatikusan ebbe a üzemmódba kapcsol, ha a helyszínt felé valamilyen élőlény közeleg (**MONSTERS NEARBY**), és villogva jelzi a ajtót, amelyen 2-3 másodperc múlva fog lépni. Ilyenkor természetesen célszerű a helyszínt elhagyni vagy átballagni a szoba másik oldalára, nehogy összeütközéssel egy csomó **STAMINA**-pontot veszítsünk.

INVENTORY ('Z' billentyű): Letár, a ablakban a erszényünkben lévő tárgyak neveinek listája jelenik meg. Ha valamilyen tárgyat elveszünk/leeszünk, a ablak automatikusan a kijelzésre kapcsol néhány másodpercra. Mint már említettük, összesen 6 tárgy lehet nálunk, többet nem tudunk felvenni, mert a program jelzi, hogy a erszény tele (**THE POUCH IS FULL**).

MAGICK ('Z' billentyű): Azoknak a varázslatoknak a listája, amire pillanatnyilag képesek vagyunk. A varázslatokat a **GRIMOIRE** varázskönyv (**BLAST**, **INVOKE**, **FREEZE**) és a két **SCROLL OF PARCHMENT** (**CALL** illetve **TRANSFUSION**) tartalmazza. Ha nincs nálunk a **GRIMOIRE**, akkor a varázslatok listájában **NO GRIMOIRE** feliratot láthatunk – és persze varázsolni nem tudunk.

SITUATION ('Z' billentyű): A helyszín nevének (**YOU ARE IN ...**), a szint számának (**ON LEVEL ...**) és varázslói besorolásunknak (**YOUR GRADE ...**) megjelenítése. Ha egy új helyszínre megyünk át, mindig ez kapcsolódik be.

BLAST ('B' billentyű): Robbantás varázslat, a **GRIMOIRE** felvétele után használhatjuk (ha nincs nálunk, a program a parancsra **YOU CAN'T NO SPELLS** feliratot jelenít meg). A **BLAST** onmagában a helyszínen tartózkodó élőlények elleni támadásra szolgál. A robbantás az ellenfél életerejét csökkenti 1-5 ponttal, a **LUCK** függvényében. Ha nincs a helyszínen ellenséges élőlény, akkor Axil szétárja a kezét és a **BLAST WHAT?** feliratot láthatunk. A robbantás irányulhat egy megadott tárgyra is, ha begépeljük a parancs után a tárgy nevét – ilyenkor Axil a megadott tárgyat próbálja szétrobbantani. Vigyázat, ha a megtámadott élőlény hirtelen elhagyja a szobát, akkor az utolsó robbantást Axil önmagán fogja végrehajtani – néhány életerőpont veszteséggel!



INVOKE ('I' billentyű): Szellemidézés varázslat, a **GRIMOIRE**-ban van. Az **INVOKE** 4 démont idézhetünk meg a nevük begépelésével, akik különböző szolgáltatásokat tehetnek nekünk. A nevük **ASTAROT**, **BELEZBAR**, **MAGOT** és **ASMODEE**. Vigyázat, a démonok megidézéséhez egy-egy talizánt is össze kell gyűjtenünk, máskülönben nincs védelmünk a hatalmuk ellen (**YOU ARE NOT PROTECTED**) és bedobnak a első szinten lévő **FURNACE** nevű kemencébe, ahol tragikus hirtelenséggel el fognak fogyni a életerőpontjaink. A démonidézésnek más megkötései is vannak, később még foglalkozunk a témával.

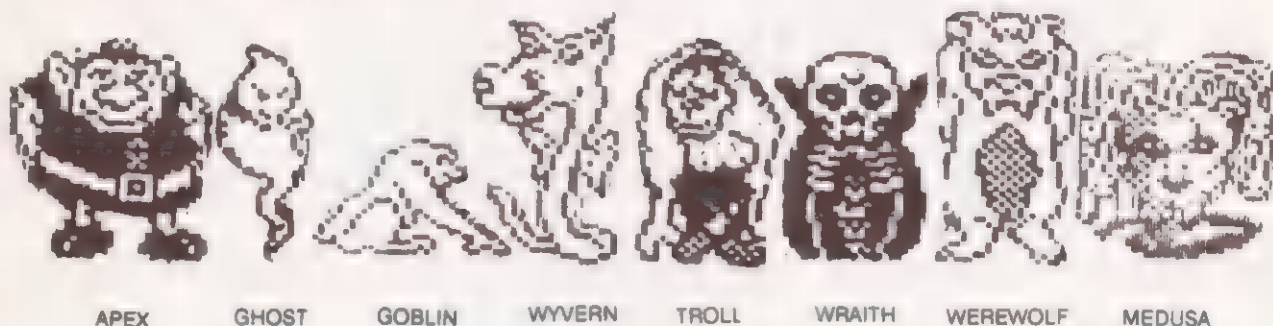
FREEZE ('F' billentyű): Fagyasztás varázslat, a **GRIMOIRE**-ban van. Meg kell utána adnunk, hogy mit akarunk fagyasztani. A képernyő ekkor villogni kezd, és a fagyasztott tárgy – a fagyasztás idejére – ugrál egy kicsit. Egy fagyasztás 4 **STAMINA**-pontba kerül. Ellenfelek ellen nem hatásos (mire begépeljük a élőlény nevét, már rég megtámadott bennünket és Axil elfelejt a parancsot), de később még hasznát vehetjük...

CALL ('C' billentyű): "Hívni valakit" varázslat. Két **SCROLL OF PARCHMENT** (varázstekercs) van a játékban (vigyázat, olyan is van, ami csak úgy néz ki!), amelyeknek felvétele után kiderül, hogy a **GRIMOIRE** lapjai (bele is épülnek, tehát nem foglalnak további helyet az erszényben) és egy-egy varázslatot tartalmaznak. Az egyik tekercsen a **CALL**, a másikon a **TRANSFUSION** varázslat van. A **CALL** segítségével a többi élőlényt hívhatjuk a helyszínre a nevük megadásával. Mivel Apex kivételével mindegyik támadni fog, célszerű csak Apex-et hívogatnunk, a többitől tartózkodjunk. Apex hívására többször is szükség lehet: ő ugyanis egy jó szándékú óriás, akivel elmés csevegést folytathatunk tárgyairól, élőlényekről, ajtókról. Néha fontos információhoz juttat bennünket. Ha a **CALL** varázslatot úgy akarjuk alkalmazni, hogy azt tartalmazó tekercset még nem találtuk meg, a program közli velünk, hogy ilyen dolog nincs (**NO SUCH THING**) – legalábbis nálunk...

TRANSFUSION ('T' billentyű): Transzfúzió varázslat, egy **CALL**-hoz hasonló tekercsen találjuk meg. A transzfúzió segítségével a tapasztalati pontokat csapóhatjuk meg, amelyek átalakulnak életerőpontokká. Erre értelemszerűen akkor van szükség, ha **STAMINA**-ból gyengén állunk. Egy transzfúzió 5 tapasztalati pontot fogyaszt el, de a **STAMINA** 10 ponttal növekedik. Nem használható transzfúzió akkor, ha egy másik élőlény van a helyszínen vagy valamilyik démon bedobott minket a **FURNACE**-be.

Az elmondottakon kívül még a kommunikáció fontos része a beszéd. Beszélhetünk élőlényekkel, tárgyakkal, ajtókkal, akadályokkal – mindennel. A program beszélni kezd mindent, amit idézőjellel kezdünk ("SYMBOL SHIFT" + "P"). Az idézőjel után azt kell begépelnünk, amihez/akihez a szózatot intézzük, majd vesszővel elválasztva azt, amit mondani akarunk neki, pl. "APEX, DOOR" ("RETURN") begépelésére a program úgy véli, hogy Apex-hez kívánunk szólni, és az ajtóról akarunk megtudni valamit. Mint a példából is látszik, két szóval "beszéd"! Hasonlóképpen kell kinyitnunk a kulcsszóval nyitható ajtókat, pl. "DOOR, SPACEBAR" (erre ugyan egy ajtó fog kinyitni, de példának megteszi). Space-t kell használnunk a vessző után! Ha szózatunk valami hatást eredményez (esetleg választ valaki vagy történt valami), akkor azt a középső ablakban szöveg jelzi, ha semmi eredmény, akkor (csend) honol a helyszínen. Előfordulhat az is, hogy olyasmire szólunk, ami nincs a helyszínen; ilyenkor NO SUCH THING (Nincs ilyen dolog) felirat jelenik meg. Egyébként magunkban is beszélhetünk, pl. "AXIL, AXIL" – ilyenkor a program elmesésen megjegyzi, hogy kezdődő elmebaj jele (IT'S A SIGN OF MADNESS). A csevegések általában roppant magas szellemi szinten zajlanak, az angol nyelvben kevésbé jártas játékosoknak nem árt, ha közük ügyében van egy angol-magyar szótár. Megjegyzendő, hogy – akár csak a parancsoknál – "DELETE" nem hagyományos módon működik: megnyomásakor FORGET IT (felejtet el) felirat jelenik meg, és a program figyelmen kívül hagyja az eddig begépelte dolgokat.

Most, hogy már tisztában vagyunk a játék kezelésével és főbb szabályaival, nem árt, ha megismerkedünk a szereplőkkel. Akárcsak a többi GARGOYLE-játékban, a NEWY THE MAGICK-ben is jónéhány élőlény próbálja megkéséríteni az életünket, akikbe vagy egy új helyszínre belépve botlunk bele, vagy ők látogatnak be arra a helyszínre, ahol túl sokat álldogálunk (már szó volt róla, hogy azt a bejáratot, amelyen valamilyen szörny fog nemsokára belépni, a program villogva jelzi). Az élőlényeknek két tulajdonsága van: a STAMINA – akárcsak nálunk – az életerőt jelenti, amelyet sorozatos robbantással (BLAST) nullára csökkentve tudjuk a páciens kiirtani; a CUNNING az ügyességüket jelzi, azaz hogy egy támadással (érintkezéssel) hány STAMINA-pontot csapolnak le tőlünk. Az egy Apex-et leszámítva, mindegyik élőlény rosszindulatú, 2-3 másodpercenként nekünk rontanak és fogyasztják életerőnket, tehát ha találkozunk valamelyikkel, akkor vagy kezdünk el villámgyorsan robbantgatni vagy távozzunk. A harc egyébként igen épületes látvány: a robbantásaink után egy kis füstelhő keletkezik az élőlény testén, a program közli, hogy "a robbantásnak volt egy kis hatása" (THE BLAST HAD LITTLE EFFECT), és a LUCK-pontszámunk függvényében 1-5 ponttal csökken az ellenfél életeréje. Természetesen ő sem marad adósunk: időről-időre nekünk ront (... ATTACKS) és ügyességének valamint szerencsénknak alapján 1-10 életerőpontot csökkent rajtunk. Ezt Axil némi kiabálással kíséri (AAAAARGH! – fordítás talán nem szükséges), valamint elfelejti az addig begépelte parancsokat. Előfordulhat, hogy harc közben az ellenfél elhagyja a helyszínt, majd 10-20 másodperc múlva megújult erővel tér vissza. Ezzel a trükkkel egyébként mi is élhetünk: ha a helyszínt elhagyjuk, majd egy kis idő múlva visszatérünk, addigra az ellenfél már általában elkötrődik onnan (hacsak nem jött közben utánunk). Ha mi győzünk (elfogynak az ellenfél életerőpontjai), a szörny eltűnik a földben (... IS DEAD) és csak egy összerombolt torzó marad belőle, valamint eredeti ügyességétől függően 1-3 ponttal növekszik a tapasztalati pontjaink száma. Ha ő győzne, a program közli, hogy szörnyű halált haltunk (YOU DIE HORRIBLY) és a játék véget ér. A győzelem záloga, hogy ne húzzunk újat a sokkal erősebb lényekkel (Werewolf-fal, Medusa-val és pláne nem Apex-szel, ő ugyanis jó szándékú), valamint a harcban minél gyorsabban robbanassunk. A különböző élőlényekkel leggyakrabban a róluk elnevezett helyszíneken találkozhatunk, pl. Troll-okkal TROLLWYND-ben, Wraith-ekkel WRAITHVALE-ben – bár ez nem törvényyszerű. Az alábbi Mosolyalbum bemutatja, hogy milyen mozgó lényekkel találkozhatunk (vannak helyhez kötöttek is):



APEX

GHOST

GOBLIN

WYVERN

TROLL

WRAITH

WEREWOLF

MEDUSA

APEX THE OGRE: Nagy melák óriás, valamilyen érdemrenddel (tán R-GO jelvény?) a mellén. Ő az egyetlen jó szándékú élőlény a játékban, ő támad, viszont vele történő összeütközés jónéhány energiapontunkba kerülhet. Természetesen ha robbantunk egyet rajta, arra reagál: hirtelen letapos bennünket, lévén életeréje 40, ügyessége pedig 50. Bár Apex rendkívül buta ábrázatot mondhat magának, azért elég sok szutnivalója van: vele csevegve néha rendkívül értékes információkhoz juthatunk. Ha találkozunk vele, gentleman módjára mutatkozzunk neki: "APEX, AXIL. "Az én vagy, te salátaagyú" – jön a válasz. Úgy látszik humoránál van, folytassuk: "APEX, APEX. "Az én vagyok" – feleli. Jó, nézzuk akkor a kollégákat: "APEX, GHOST. Nem hisz a szellemekben, mert azt mondja, hogy "nincs ilyen dolog". Őké, akkor viszont mi a véleményed a Troll-okról: "APEX, TROLL. Úgy véli, hogy "egy troll-t a legjobban megölni" (tényleg!). Akkor nézzünk valami nagyobb kaliberű ellenfelet: "APEX, WEREWOLF. Azt mondja, hogy "a hagyományos módszerek a legjobbak" (tipikus GARGOYLE-féle információ!). Váltunk témát: mi a véleménye mondjuk a GRIMOIRE-ról ("APEX, GRIMOIRE). Azt mondja, mutassuk meg neki (SHOW ME THE GRIMOIRE). Ha azt mondja, hogy a kért tárgy mutassuk meg neki (azaz tegyük le a földre), akkor majdnem biztos, hogy értéktelen információt fog mondani, például azt, hogy bizonyosan az, ami neve (IT'S CERTAINLY LOOKS LIKE ...), bár a GRIMOIRE letétele után azt mondja, hogy "ez egy Grimoire, néhány megjegyzéssel". Marha. Na, húzzál innen! ("APEX, THANKS) "Örülök, hogy segíthettem!" – mondja, és eltűnik.

Mint az iménti példákban is kiderült, mindenféle berendezési tárgyról, élőlényről, jelről, felvehető tárgyról el lehet beszélgetni Apex mesterrel. Bár a példákban az okosságainak tárházából csak kevésbé részesültünk, néha kiváló segítséget szolgáltat. Kézenfekvő tehát, hogy ha valamilyen ismeretlen dologra bukkantunk, akkor hívjuk magunkhoz Apex-et és kezdünk vele beszélgetni róla. Ha a tárgy neve után azt mondja, hogy NOTE, akkor a tárgyat valamilyen fel tudjuk használni! Egyébként vannak kézzelfoghatóbb információi is, viszont csak a dolgokról hajlandó beszélgetni velünk, ha elvont fogalomról (pl. kulcsazóról) vagy nem létező tárgyról beszélünk, azt mondja, hogy nincs ilyen dolog (NO SUCH THING) – bár a szellem példája mutatja, hogy attól még lehet. Tárgy esetében előfordulhat, hogy azt kéri, mutassuk meg neki (SHOW ME THE ...) – ilyenkor tegyük le a földre és kérdezzük ismét. Ha valamilyen tárgyról megállapítja, hogy ez bizonyosan az, aminek látszik, tud semmit a dologról, ne is fárasztuk vele. Mikor megelégtettünk a beszélgetéssel, köszönjük meg a fáradozásait, szépen elkötrődik. Egyelőre ennyit Apex bonyolult lelkivilágáról, későbbiekben még sokszor találkozunk vele.

GHOST: Szellem, 6-os életerővel és 18-as ügyességgel. Nem ellenfél.

GOBLIN: Csimpánz-szerű, ide-oda mászkáló kreatúra. Életerej 6, ügyessége 20, tehát őt sem túl nehéz legyőzni.

WYVERN: Ez valami sárkányfióka lehet, aki rendkívül hülyén vigyorog. Az előbbi kettőnél jóval keményebb és gyorsabb ellenfél: életerej 10, ügyessége 25. Elpusztítása viszont nem csak 1, hanem 2 tapasztalati pontot eredményez.

TROLL: Félmeztelen púpos alak, bunkóval a kezében. Ő a leggyengébb szörny: az életerő 6, a ügyesség 15. Kevés...

WRAITH: Csontváz fekete köpenyben. Wyvern-szintű ellenfél: életerő 10, ügyesség 18. Elpusztítása 2 tapasztalati pont.

WEREWOLF: Dühösen topogó farkasember, akivel azután találkozhatunk, miután kinyitottuk a farkasok által őrzött ajtót. Rossz tulajdonsága, hogy néha cseppkőoszlopnak álcázza magát, majd hirtelen megjelenik. Ő egyébként már kemény legény, nem tanácsos ujjat húzni vele – legalábbis a robbantással (van egy tárgy, ami megvéd tőle). Életerő 20, ügyesség 25.

MEDUSA: Egy lábasfejű holgy, akinek a arca pepitára van sminkelve, a fején pedig ide-oda lobogó gumiekszereteket visel. Apex-szintű nehézsúlyú szörny: életerő 40, ügyesség 50. Kár is robbantgatással próbálkozni nála, csak vesztesek lehetünk. A kalandjátékban szereplő medúzák arról híresek, hogy pillantásukkal kővé változtatják a halandó emberiákat. Ez a holgy mondjuk nem csinál ilyeneket, viszont elég ronda – vajon mit szólna, ha egy tukorben meglátná magát?...

VAMPIRE: Vámpír. Életerő 40, ügyesség 50. Tulajdonságai azt mutatják, hogy nem érdemes vele ujjat húzni.

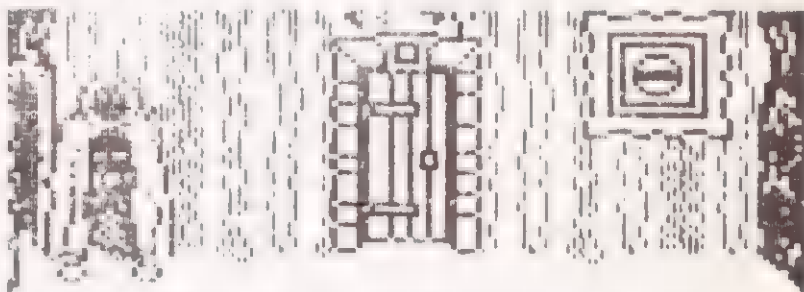
SLUG: Ronda malacpofa, pókhassal és zakóban. Életerő 50, ügyesség 35.

Az utóbbi két szereplő már a fent a Mosolyalbumba – mindenki megismerkedhet velük személyesen. Az utolsó négy élőlény csak bizonyos helyeken szokott előfordulni, ahova általában egy zárt ajtó kinyitásával kecmereggetünk át. Mint a tulajdonságaik is mutatják, egyikkel sem érdemes harcba keveredni, csak akkor, ha nincs más választásunk (például elállják az egyetlen kijáratot). A harc azért is felesleges, mert mindegyik ellen védekezhetünk egy-egy kabalával, ha a hatásos kabala nálunk van, a ellenfél azonnal szörnyethal, mielőtt nekünk ront (bár a nem eredményez tapasztalati pontot). A kabalákról részletesen a tárgyak ismertetésénél beszélünk.

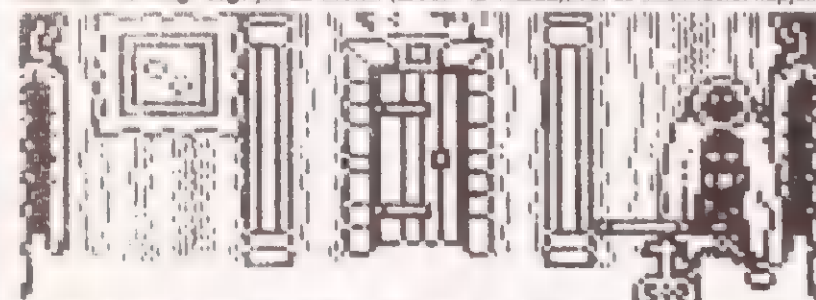
No, ezek lettek volna a szereplők. Aki a játékot ezek után egyedül kíváná végigjátszani (bár kételkedünk benne, hogy lenne ilyen mazo-chista), hagyja abba itt a leírás elolvasását, készítsen maga mellé egy nagy adag idegnyugtatót és vegyen ki legalább fél év fizetésnélküli szabadságot (bár lehet, hogy kevés lesz). Célszerű előbb ismerkedni a tereppel (talán a térképmelléklet segít valamit), sok tapasztalati pontot összegyűjteni, megkeresni a GRIMOIRE másik két lapját, és sok játékállást menteni (keveset káromkodni!). Rajta...

A pedig megkezdjük a megoldás ismertetését – azon a módon, amit a GARGOYLE-játékoknál eddig is követünk: nem lépésről-lépésre (úgy 8-10 számot is elfoglalna), hanem az egyes rejtélyek megoldásával illetve a tárgyak lelőhelyével és funkciójával. A menetrend szabadon választott. Először is a ajtókra kerítünk sort. Egy kis mászkálás után feltűnhet, hogy néhány kijárat nem a szokásos barlang-száj alakú, hanem szabályos ajtó, amely mellett néha valamilyen jelölés is található. Természetesen zárva vannak (a térképen a átjáró áthúzással jelölve). Ha Axil-tal egy ilyen kijáratot próbálunk használni, tanácstalanul széttárja a kezét és közli, hogy zárva van (LOCKED). A zárt ajtóknak négy fajtáját különböztethetjük meg: pénzre, kulcsszóra, kulcsra vagy egyáltalán nem nyíló ajtók. Speciális eset még a FURNACE, ahol nincs is ajtó: kijönni nem lehet, csak meghalni. Nézzük sorban a zárt ajtókat:

Számos olyan bezárt ajtót találhatunk, amely mellett egy kör alakú jel látható, benne egy fekete vonallal. Ha idehívjuk Apex-et, és megkérdezzük mi a véleménye a ajtóról ("APEX, DOOR), azt feleli, hogy kijárat Axil-nak (WAY OUT TO AXIL). Hm, hát ezzel a sokat segítettél te nagy melák... Esetleg a jelről tudna referálni ("APEX, SIGN). Azt mondja, ez egy "nem bejárat" jelzés (IT'S A NO ENTRY SIGN). Aha, szóval ez egy inverz "behajtani tilos"-tábla. Az ilyen jellel jelzett ajtókon – erről a oldalról – nem tudunk átkelni, csak valamelyik ajtó kijáratként funkcionálnak.



A következő zárt ajtótípus mellett jeleken két kört láthatunk egymás alatt. Feltűnő érdekesség, hogy az ajtók mellett egy asztalka is található. Ha megvizsgáljuk a asztalt (EXAMINE TABLE), azt az információt kapjuk, hogy ez egy asztal egy kulcs részére (IT'S A TABLE FOR A KEY).



Forduljunk segítségért ismét Apex-hez: "APEX, KEY. Közli velünk, hogy mutassuk meg neki azt a kulcsot (SHOW ME THE KEY). Jaj, te szerencsétlen, mi is azt keressük! Mindegy, talán a jelről tud valamit nyilatkozni ("APEX, SIGN). Közli velünk, hogy a egy váms jelzés (IT'S A TOLL SIGN). Aha. a váms? ("APEX, TOLL). Nincs ilyen dolog – jön a válasz. Okos vagy, Api. Talán nem kell sok fantázia hozzá, hogy egyedül is kitaláljuk: a váms a jónéhány helyen megtalálható aranyzacskók (BAG OF GOLD – vigyázat, néhányuk megmérgezővel) képeiben testesül meg. Ha

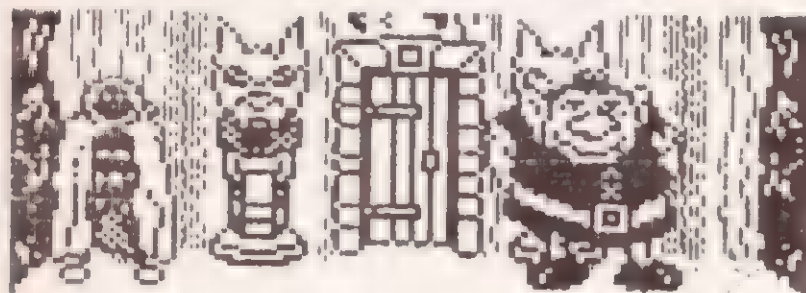
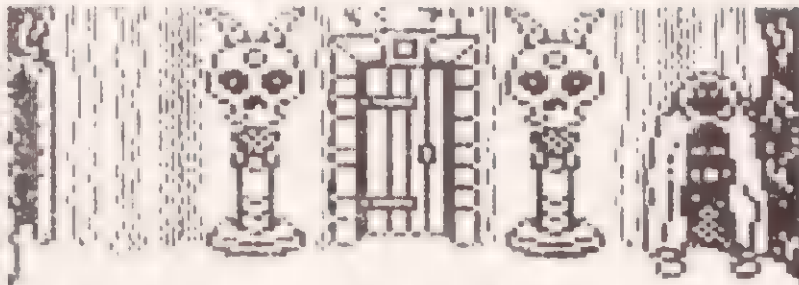
egy ilyen zacskót leteuszunk a asztalra (csak simán a földre nem jó!), akkor a ajtó egy CLICK! felirat kíséretében kinyílik. Természetesen a zacskót elveszszük a asztalról, akkor a ajtó is rögtön bezáródik. A váms átjárók másik kijáratja természetesen nem váms, tehát előfordulhat, hogy egy déli/délkeleti/délnyugati irányú átjárón bemenve egy olyan ajtón jövünk ki, ami tulajdonképpen zárva van, innen csak vámmal nyitható. Mindenesetre a árt, ha egy zsák aranyat mindig magunknál tartunk.

Az ajtók harmadik típusát képezik azok, amelyek egy bizonyos kulcsszó kimondására (pl. "DOOR, SHAZAM) nyílnak ki. Ezeket megismerhetjük onnan is, hogy a ajtók mellett lévő oszlopok tetején valamilyen állatfej (farkas, kecske, stb.) vannak. Ha megvizsgáljuk őket (EXAMINE OBJECT), kiderül róluk, hogy ezek őrszlopok (PILLAR OF GUARD). Szólítsuk magunkhoz Apex-et, és érdeklődünk nála, hogy mi a véleménye a ajtóról ("APEX, DOOR). Azt mondja, hogy kijárat Axil-nak. Hm, ezt egyszer már eljátszottuk valunk, kár volt megint megkérdezni tőle. Akkor tudakoljuk, hogy mit tud a őrről ("APEX, GUARD). Közli velünk, hogy ilyesmi nem létezik. De hát itt vannak előtted, te ló! Tényleg, a – vagyis többes számban kell beszélni róluk (velük). Beszélgetésünk akkor velük is egy kicsit: "GUARD, DOOR kérdésre például valami kis segítséget nyújtanak az ajtót nyitó kulcsszó megfigyeltéhez. Ezek természetesen leginkább a szokásos GARGOYLE-féle segítséget jelentik, a legtöbbszor csak akkor lesznek értelmesek számunkra, amikor már amúgy is kitaláltuk őket. A nagyobb baj azonban, hogy egy ilyen ajtó kinyitására (tisztelőt a kivételnek) egy csomót kell mászkálnunk, egyes helyekre való eljutáshoz jónéhány tárgyat meg kell találnunk és jól felhasználnunk – és mindezek közben nem árt életben is maradnunk.

Az alábbiakban sorban ismertetjük ■ kulcsszavas ajtók kinyitásának módját. A kulcsszavas ajtókat a térképen A – G-ig jelöltük, ■ helyszínekre ■ hely névvel, illetve a szektorszámmal fogunk hivatkozni. A szektorszám első száma ■ szint számát jelzi, a második és harmadik ■ X- (balról jobbra) és Y (felülről lefelé) koordinátákat – ■ koordináták értelmezéséhez egyébként segítséget nyújt az 1 szint térképe melletti számozás is). Ennek megfelelően, a startszoba szektorszáma 246, míg ■ FURNACE a 181.

Bár ■ későbbiekben egyenként is ismertetjük minden egyes tárgy használatát is, ■ ajtók kinyitásának ismertetésénél is – ha szükséges – kitérünk néhány tárgy használatára is – további információ róluk később, a tárgyak részletes ismertetésénél).

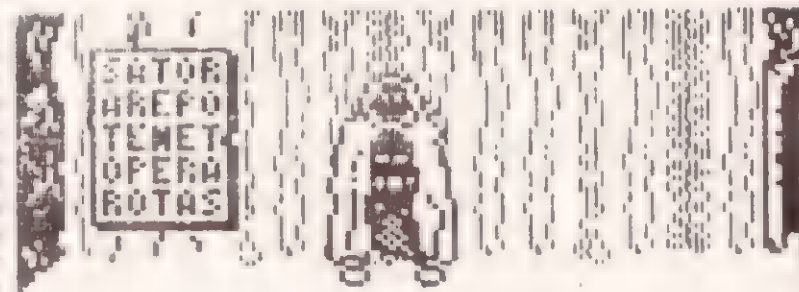
Az A-val jelölt szoba a SECUNDA PORTA-ban van (284). Az örökkel konzultálva ("GUARDS, DOOR") tudomásunkra jut, hogy az ■ bizonyos szó nem ■ szó. Hát bizony ez egy elég épületes megállapítás... Akkor mi ■ ördög lehet: mondat? "DOOR, SENTENCE – semmi sem történt, azonkívül, hogy klródott ■ LOCKED (zárván) szó. Talán esetleg egy egész könyv?" "DOOR, ■■■■ – még mindig semmi. Talán rossz irányba tapogatózunk... Próbáljunk egy kicsit elvonatkoztatva gondolkodni ■ dolgokon: rájöhettünk, hogy az ■ szó, ami nem is szó, ■■ talán el ■■■■ hangzik igazából. Akkor tan nem is beszéd, hanem csak valamilyen zaj. "DOOR, NOISE – semmi. Talán csak egy hang? "DOOR, SOUND – még mindig semmi. Simon és Garfunkel egy száma szerint azonban van egy olyan dolog, aminek van hangja – bár olyan mintha nem lenne. Ez pedig – ■ csend. "DOOR, SILENCE CLICK! – mondja az ajtó és kinyílik. A mögötte levő helyszínen ugyan nincs semmilyen fontos tárgy vagy élőlény, és kijárat is csak visszafelé vezet. ■ helyszíne belépve egy igen lényeges dolog történik: ■ fejünkben tomboló asszociációs mámor díjazásaképpen varázslói besorolásunk NEOPHYTE-ről ZELATOR-ra (2:9) emelkedik. Ez nagyon fontos pl. szellemidézésnél, BELEZBAR szolgálatásait például csak ZELATOR-szintű vagy annál jobb varázsló veheti igénybe.



nevezését (WOLFDORP = "Farkasfalva"), ■ oszloptól levő állatfejet meg azt, hogy ez az állat elég gyakran szokott "kiabálni", ■■ ordítani. Bizony, bizony: ő ■ Piroška-burger nagy hódolója – ■ farkas. "DOOR, WOLF – és ■ ajtó egy kattánással feltárl. Nem árt, ha ■ WOLFDORP-ban maszkálva óvatosak vagyunk: itt ugyanis minden helyszínen várható ■■ kedves WEREWOLF feltűnése, aki egyébként egyike ■ legkellemetlenebb ellentelnek. Megjegyzendő, hogy előszeretettel álcázzák magukat cseppkőnek – ne csodálkozzunk tehát azon, ha ■ szörny közeledését jelző műszer nem jelez, mégis egyszer csak egy idegesen topogó WEREWOLF néz velünk – szó szerint – farkaszesmet. A hagyományos robbantató harci taktikával nem nagyon tudjuk legyőzni, viszont egy ezüströg (NUGGET) nevű talizmán segít ■■ ugyan: a farkasemberek kimondottan utálhatják ezt, mert mihelyt nekünk rontanak, azonnal elhaláloznak. Természetesen ■ WEREWOLF elleni talizmán megszerzése is némi akadályokba ütközik – erről majd később lesz szó.

A C-vel jelölt ajtó szintén WOLFDORP-ban található, csak egy ideig tekeregnünk kell addig, amíg elerünk oda (174). Az öröknél ■ ajtó-ról érdeklődve azt a választ kapjuk, hogy ■ belépés ide hülyeség (TO ENTER IS MADNESS). Hogyne volna az – már ■ HEAVY ON THE MAGICK-be is az volt... A kinyitáshoz semmi más segítséget nem találtunk – kénytelenek voltunk tehát egy kicsit (egy nagyot) gondolkodni. Végző kétségbeesésünkben megpróbáltunk ■ MADNESS szó szinonimáival szórakozni: ez végre céliravezető megoldás volt, ugyanis kiderült, hogy ■ keresett szó ■ LUNACY.

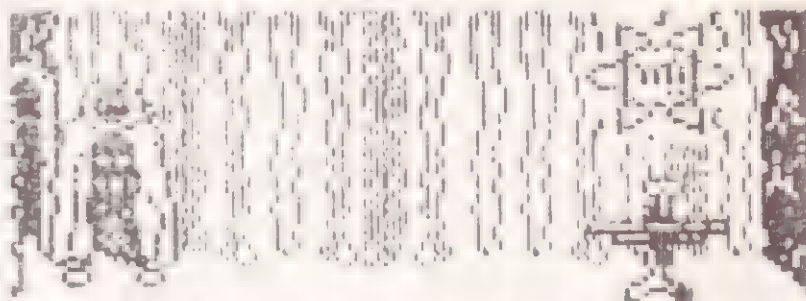
A D jelű ajtó ■ QUADRA PORTA-ban van (258). Ide eljutni már egy kész Kék-túra (már amennyiben nem rendelkezünk kard (SWORD) nevű talizmánnal – ezzel ugyanis megidézhetjük ASTAROT demont, aki tudvalevőleg ■ teleportálás nagymestere): TROLLWYND-ből ROOM OF FLOX-ba átmenve ki kell nyitnunk az ajtót ■ megfelelő kulccsal; ekkor átjutunk SLYMOLE-ba, amelynek utolsó szobája ■ ROOM OF PURITY, ahol egy újabb kulcsos ajtó vár ránk; miután ezt is kinyitottuk, végre átkerülünk ■ QUADRA PORTA-ba, amelynek utolsó szobájában rálehetünk ■ nyomorult ajtóra. Az ajtóról tudakozódva ■ örök közlik velünk: THE GREAT ■■ I IN FREE. Lehet, hogy egy John Smith vagy egy Richard Davis nevet viselő úr (lévén valószínűleg angol anyanyelvűek) bizonyára meszemenő következtetéseket vonhat ■ ebből – nekünk azonban nagyon meggyűlt ■ bajunk ■ fordítással: talán "a nagy jel én vagyok, szabadon" vagy – lévén az I éppen úgy ejtendő, mint ■ EYE (szem) – "a nagy jel szem szabadban"? Lehet, hogy itt kell kijutni ebből ■ ronda barlangrendszerből? Vagy teljesen mást jelent? Mindegy, ■ lényeg az, hogy kiderült, ■ meglehetősen számos jel közül a startszobától balra levő helyszínen (SOTHIC COMPLEX – 236) látható dologra gondolni. Első ránézésre úgy néz ki, mint egy SpV.-keresztjeztvény: ■ füle, ■ farka. Észrevehető azonban, hogy ■ jelben lévő sorokat bármilyen irányban olvasva, visszafelé is ugyanazt kapjuk (Rózsa Gyuri biztos tudná rá egy spéci kifejezést, de nekünk most éppen nem jut eszünkbe). Lehet próbálkozni mindenféle furfanggal, pl. lóugrásban összeolvasni ■ betűket, csigavonalban belülről kifelé és kívülről befelé, csak minden másodikat, minden harmadikat és így tovább. Mindegyik kiváló ökörségeket eredményez – de egyik sem céliravezető. Egyébként ez nagyon rossz vicc volt ■ játék "elkövetőtől": megoldás kulcsa ugyanis innen (az ajtótól meg pláne) fényévekre található, ■ H jelű ajtó mögött (427), ráadásul egy rubinba vésve. A véset úgy hangzik, hogy WONTOOTOO. Ez így elég épületesen néz ki, viszont fonetikusán megegyezik azzal, ha valaki angolul azt mondja: 122. Namármost, ha ■ 1-et "igaz"-nak



A B jelű ajtó WOLFDORP első szobájában van (144). Az örökkel ■ kulcsszó titkáról beszélgetve ("GUARDS, DOOR") azt ■ választ kapjuk, hogy CRY AND ENTER IT. Az ■■ ENTER IT jelentése nyilvánvaló: "■ és lépj be". A CRY azonban egyaránt jelent sirást és kiáltást, kiabálást is. Eláruljuk, hogy ■ nyomorult ajtónak akármit sirhatunk – nem fog kinyílni. Tehát kiabálni kell: "DOOR, SHOUT, "DOOR, CRY vagy "DOOR, HOWL egyaránt hatás-talan – megint egy kis asszociációra van szükségünk. A kulcsszó kitalálása egyébként nem lesz túl nagy nehézség, ha figyelembe vesszük ■ hely el-

vesszük. ■ 2-t pedig "hamis"-nak és a bal felső (vagy a jobb alsó – teljesen mindegy) saroktól kezdve (balról jobbra) ■ módszer alapján olvassuk össze az "igaz" karaktereket, megkapjuk ■ ajtó kulcsszavát. Ez talán már mindenkinek sikerülni fog – egyébként érdekes módon ez is ugyanúgy hangzik visszafelé olvasva is, mint eredetileg. A szobába belépve érdekes történés zajlik le, amire ■ leírás végén (most már kezd úgy tűnni, hogy az ■ következő SpV-ben lesz...) még visszatérünk.

Az utolsó három kulcsszavas ajtó kinyitásának módját most egy időre elnapoljuk – ezek már nagyon bonyolult cselekménysorozatok, ahol viszont feltétlenül szükség van a kulcsok által nyitható szobák kinyitására (eddig is történt már rájuk utalás). A kulccsal nyitható ajtók pont olyanok, mint ■ vámos ajtók, csak nincs mellettük ■ ilyen jelzés. Az ajtók mellett egy-egy asztal található, ahol ■ megfelelő kulcsot elhelyezve, az ajtó kinyílik. A játékban 12 ilyen ajtót nyitó kulcsot lehetünk. A kulcsot tartalmazó helyszíneken a falon egy jel található, mindegyik egy nap jelbe beleerajzolva. Megint ■ kis asszociációs mámorra ■ szükségünk ahhoz, hogy kitaláljuk ■ napocskába rajzolt jelek ■ csillagövi jegyek stilizált megfelelői – bár néha nem teljesen egyértelmű, hogy a jel mit is akar ábrázolni. Mindegy, ha sikerült kiszűri, hogy mi is lehet az, meg kell keresnünk ■ hozzá tartozó ajtót. Az ajtó és ■ kulcs között a kapcsolatot ■ szobának ■ ■ jelenti, ahol ■ ajtó van (pl. ■ vízöntő csillagövi jegynek megfelelő jelnél felvett kulcs nyitja ■ Esők Szobája helyszínen lévő ajtót). A kulcsok megvizsgálásakor kiderül róluk, hogy milyen anyagból vannak, bár ez igazából nem lényeges. Mivel – mint említettük – ■ kulcsokat jelölő ábrák meglehetősen stilizáltak, nem árt, ha felsoroljuk, hogy melyik ábra mit jelent:



Nyílás
CHROMA KEY



Mérleg
BRASS KEY



Halak
COPPER KEY



Ikek
LITHIC KEY



Szűz
ALUMINIUM KEY



Kos
MAGNET KEY



Oroszlán
NICKEL KEY



Bika
IRON KEY



Bak
BRONZE KEY



Rák
TIN KEY



Skorpió
ZINC KEY



Vízöntő
COBALT KEY

CHROMA KEY: krómkulcs, WOLFDORP-ban van (121). Mivel ■ nyílás csillagkép tartozik hozzá, ■ Nyílak Szobáját (ROOM OF ARROWS) nyitja (157).

BRASS KEY: sárgaréz kulcs, ROOK OF HYDRA-ban van (346). Talán nem lesz nehéz kitalálni, hogy ■ Skála Szobáját (ROOM OF SCALE) nyitja (424), mivel ■ mérleg jelöli.

COPPER KEY: vörösréz kulcs, SOTHIC COMPLEX-ben van (255). halak csillagképpel jelölve. A ROOM OF ICHTYS-t nyitja (233), ■ következő összefüggés miatt: ■ hal az ókori rómaiak által üldözött ökeresztények vallási szimbóluma volt, görögül ■ neve úgy hangzik: ICHTYS. Keresztény szimbólum az éret szolgált, mert a Jesus Christus Theon Yisos Sother (Jézus Krisztus, Isten fia, Megváltó) szöveg kezdőbetűi összeolvasva ezt ■ szót adják. Bővebb felvilágosítás *Slendewicz: Quo* ■ c. könyvében (HEAVY ON THE MAGICK-leírás helyett ajánljuk...).

LITHIC KEY: lítiumkulcs, WRAITHVALE-ből (261). Az ikek csillagkép jelzi – bár ■ ábra szerint már rögtön szíami ikek – és ■ ROOM OF TWO ON helyszín ajtaját nyitja (327).

ALUMINIUM KEY: Alumíniumkulcs, WRAITHVALE-ben lehetünk rá (281). Meglehetősen érdekes ábra tartozik hozzá, de később kiderül, hogy ■ szűz csillagképre gondoltak, mivel ■ Tisztaság Szobáját (ROOM OF PURITY) nyitja (238).

MAGNET KEY: Magnéziumkulcs, TROLLWYND-ben (351). Kos csillagkép és – bár fogalmunk sincs miféle összefüggés miatt (talán kizárólagos alapon) – ROOM OF NANI (336) ajtaját nyitja. Esetleg volt GARGOYLE-eknek egy NANI nevű hím birkájuk...

NICKEL KEY: Nikkelkulcs, SOTHIC-ban van (276). Bár ■ ábrából nem kimondottan lehet ráismerni, ■ oroszlán csillagjegyre tartozik és ennek megfelelően ■ Büszkeség Szobáját nyitja (447).

IRON KEY: Vaskulcs, METHOS-ban találjuk meg (483). A bika csillagképhez tartozik, és a Szarvak Szobáját (ROOM OF HORNS) nyitja (225).

BRONZE KEY: Bronzkulcs, GORBURG-ban botlunk bele (321). A bak csillagképet találjuk mellette, ezért ■ Nyáj Szobáját (ROOM OF FLOX) nyitja (244).

TIN KEY: Ónkulcs, MORFANG-ban van (114). Bár jóézésű ember a mellette levő ábrát halaknak vagy esetleg ikeknek nézné – épp ezért ez ■ rák. A kulcs ■ Karmok Szobáját (ROOM OF CLAWS) nyitja (158).

ZINC KEY: Cinkkulcs, WOLFDORP-ban található (173). Védjegye a skorpió, tehát ■ Fullánkok Szobáját (ROOM OF STINGS) nyitja (136).

COBALT KEY: Kobaltkulcs, TROLLWYND-ben találjuk meg (364). A vízöntő jelöli, ezért ■ Esők Szobáját (ROOM OF RAINS) nyitja (368).

(Mindenki őszinte rémületére közöljük, hogy a következő számban FOLYTATJUK!)

SPECTRUM programok átírása 5.



Elszántabb olvasóink okulva ■ előző részben leírtakból, már bizonyára betekintést nyertek ■ nevezetes 256 byte hosszú **LOADER** lelkivilágába. Aki még – kellő önbizalom hiányában – visszaríadt ettől ■ lépéstől, annak megvilágítjuk ■ probléma hátterét. Mielőtt belemélyednénk, szeretnénk néhány tanácsot adni:

■ Az **ASMON**-ban ■ gépi kódú programok betöltésére ■ "R" kimentésére ■ "S" parancs szolgál. Mindkettő rákérdez a ■ és ■ végcímre, valamint ■ betöltendő file nevére. A **SPECTRUM** programok átírásánál gondot okoz, hogy **SPECTRUM**-on ■ RAM terület ■ és ■ között helyezkedik el, míg ■ csak ■-től ■-ig tud file-okat betölteni. A probléma abból származik, hogy ilyen felállásban ■ abszolút címhivatkozások ■ érvényesek, nehezebb megtalálni ■ egyes szubrutinokat. ■ címek megtalálásának könnyítése érdekében érdemes 4000H-val alacsonyabb címre tölteni ■ programot. Ebben ■ pl. ■ "CALL ■" utasítás által hívott szubrutin kezdőcíme esetünkben ■ 7800H lesz, hanem ennél 4000H-val kevesebb, vagyis 3800H. Természetesen nem ■ egyedüli és üdvöztető megoldás, viszont ■ későbbiekben ilyen módszerrel fogjuk ismertetni ■ átírás egyes fázisait.

■ Az "R" parancs végrehajtása után a monitor ■ egy "Last address" üzenetet, valamint egy címet. Ezt ■ címet érdemes felírni, ■ kimentéskor ■ cím lesz ■ végcím.

● Ha kazettás rendszerben dolgozunk, a javított programrészeket ne ■ előző változat helyére mentjük ki. Az így előálló többletkereséseket ■ több kazettás módszerrel védhetjük ki, nevezetesen, külön kazettán legyen ■ forrászöveg, ■ SCREEN, ■ program. Ez annyi kényelmetlenséget okoz, hogy kipróbáláskor cserélni kell ■ kazettákat, ■ ez mégis a kisebb baj. Ha lemezes rendszerünk van, akkor mindig készítsünk biztonsági másolatot, mivel ■ floppy ■ az eszköz, ahol kevés munkával igen sok adatot lehet elrontani. Ez csak ■ egyik ok. Előfordulhat olyan is, hogy ■ általunk kijavított program nem azonosul ■ feladatával, és utána nem is tudjuk visszajavítani ■ javítást. Ekkor kell visszamenni az előző, kevésbé jó, de még üzemképes változathoz.

Ennyi ■ leírás után térjünk rá ■ lényegre:

Töltsük be ■ eredeti helyére a **LOADER**-t (ez ■ kivétel ami a szabályt erősíti). Az eredeti hely esetünkben B3B0, ez még ■ szabad memória területen belül van. ■ betöltött programot listázzuk ki ("L" parancs). Mivel tudjuk az indítási címet, érdemes itt próbálkozni (aki kőkemény egyéniség, természetesen próbálkozhat máshol is, de kijelentjük, hogy igazat szöltünk). Az indítási cím esetünkben adódik magától, mivel megegyezik ■ betöltési címmel. Ha elkezdjük listázni, akkor ■ következő látvány tárul ámuló szemünk elé:

```

31 FF 5B LD SP,5BFF
; A verem beállítása
B3B3 3E FF LD A,FF
; Annak jelzése, hogy nem fejléc következik
B3B5 37 SCF
; A töltés jelzése (ha CY=0, akkor VERIFY)
B3B6 00 21 00 40 LD IX,4000
; Blokk kezdőcíme
11 LD DE,1800
; Blokk hossza
CD E1 B3 CALL B3E1
; LOAD
B3C0 3E FF LD A,FF
B3C2 37 SCF

```

```

B3C3 00 21 C4 E8 LD IX,E8C4
; ■ lényegét lásd fent!
B3C7 11 3C 17 LD DE,173C
B3CA CD E1 B3 CALL B3E1
B3CD 3E FF LD A,FF
B3CF 37 SCF
00 21 76 77 LD IX,7776
B3D4 11 A4 38 LD DE,38A4
B3D7 CD E1 B3 CALL B3E1
B3DA 31 77 E6 LD SP,E677
; A verem bállítása (ismét)
B3DD F3 DI
; Megszakítás tiltása
B3DE C3 E8 EA JP EAE8
; Indítás
B3E1 14 INC D
; Ez itt ■ LOAD szubrutin

```

Mielőtt valaki ■ szavahihetőségünket kétségbe vonná, szeretünk le- szögezni, hogy ■ megjegyzések, valamint ■ logikai egységek szétválasztása a mi merényletünk. Látható, hogy ■ **LOADER** 6 logikai egységből áll. Az ■ ■ normál vagy mezei LD SP,5BFF utasítás. Ennek ■ verem állításán kívül semmi szerepe sincs. A második egység már érdekesebb. ■ látjuk, hogy 4000H-ra töltődik, ráadásul 1800H a hossza, besorolhatjuk ■ **SCREEN** skatulyába. A konvertálására ■ előző epizódban leírt játékszabályok érvényesek.

A ■ és ■ negyedik egység végzi ■ munka oroszlánrészét: ők töltik be ■ tulajdonképpeni programot, ráadásul két részletben. Az egyik E8C4H-ra 173CH, a másik 7776H-ra 38A4H mennyiségű byte-ot tölt.

Az ■ egység végzi ■ program végleges elindítását, érde- ■ megfigyelni, hogy EAE8H-n indul ■ **MOONCRESTA**.

Végül ■ ■ egység, ■ **LOAD** szubrutin, amely ■ kazettáról betölti ■ egyes file-okat. Tévedés ■ essék, ez nem egy sorból áll, sőt ■ leghosszabb szubrutin ■ **LOADER**-ben, viszont nem láttuk értelmét teljes egészében közölni.

Akit érdekel, az tanulmányozhatja, nincs benne semmi extravagáns.

Ha valaki ennyi balsezerencse és sok-sok vizsály után már a hetedik menyországban érezné magát, ■ ki kell ábrándítani. ■ **cracker** nagy showman lehetett, mivel még ■ kis meglepetést tartogat ■ tarsolyában.

Töltsük ■ két modult! Az elsőt A8C4H-ra (ez már a 4000H-val alacsonyabb cím!), ■ másodikat 3776H-ra. Az első BFFFH-ig tart, ■ második 7018H-ig, erre ■ kimentésnél legyünk tekintettel! Listázzuk ki ■ programot ■ indítási címtől kezdve!

```

AAE8 3E 00 LD A,00
AAEA D3 FE OUT (FE),A
; A keret (BORDER) fekete
AAEC CD C4 E8 CALL E8C4
; Ellenőrző byte ■ SCREEN-ből
AAEF 21 D6 LD HL,E8DE
; Összehasonlítja ■
AAF2 BE CP (HL)
; memóriarekesz tartalmával
AAF3 CA 01 JP Z,E801
; Ha egyezik, E801H-ra (A801H) ugrik
AAF6 21 48 EE LD HL,EE48

```


A SpV. 16. számának térkép-mellékletén már találkozhattunk ezzel a játékkal. Most megpróbálunk néhány – a játék teljesítéséhez szükséges – hasznos információt közölni, ugyanakkor a mostani térkép-mellékleten vázlatosan bejelöltük azt, hogy melyik varázslót hol találjuk meg. A szám a varázsló számát jelöli.

A SORCERY c. játékot az ENTERSOFT egyik híres programozója, Joy Broadhead dolgozta ki ENTERPRISE gépre, 1985-ben. A grafikája színvonalas, a zenéje kellemes. Itt szeretnénk azt is megjegyezni, hogy a kazettatokban lévő leírás kissé hibás, ugyanis a tájékoztató szerint 3 varázslót a kiszabadítunk a sötét szellemidéző fogságából, a valóságban viszont 8-at!

Az 1. varázsló kiszabadítása:

A varázsló az ugyanazon pályán található könyv (spell book) segítségével szabadítható ki. Szálljunk rá a varázslót bezáró 'dugó'-ra, magától kinyílik, majd szálljunk rá a varázslóra is, és ezzel megtörténik a szabadulás.

A 2. varázsló kiszabadítása:

Ha itt vagyunk, menjünk jobbra fel, balra fel, balra fel, majd ismét jobbra fel, és már is vagyunk a varázslónál. Szálljunk rá a varázslót bezáró 'dugó'-ra, ez eltűnik, majd szálljunk rá a varázslóra...

A 3. varázsló kiszabadítása:

Ha már ezen a pályán vagyunk, akkor menjünk jobbra, jobbra, jobbra, jobbra, jobbra, jobbra le, jobbra fel, majd vegyük fel az üveget (large bottle), induljunk el jobbra fel, menjünk oda a serleget bezáró ajtóhoz, ez ki fog nyílni. Vegyük fel a serleget (goblet of wine), menjünk vissza (balra fel, jobbra le, balra le, balra fel, balra fel, balra fel, balra le, balra le), ezután már megszokott módon menjünk oda a varázslót bezáró 'dugó'-hoz, ami magától kinyílik, és menjünk rá a varázslóra...

A 4. varázsló kiszabadítása:

Menjünk jobbra, jobbra, jobbra le, vegyük fel a kulcsot (door key), majd menjünk jobbra le, jobbra fel, jobbra fel, ezután menjünk rá arra a 'dugó'-ra, amelyik egy 'holdat' (sorcerer's moon) zár be. Vegyük fel a holdat, menjünk balra fel, balra fel, balra le, balra le, balra le, ezt követően menjünk oda a varázsló ajtajához, ami automatikusan ki fog nyílni, s menjünk rá a varázslóra...

Az 5. varázsló kiszabadítása:

Menjünk balra, a vízész van egy címer (coat of arms), vegyük fel, menjünk vissza a szárazföldre (vigyázzunk, nagyon könnyen belepottyanhatunk a vízbe, amit varázslónk nem nagyon szeret, mert minden fürdőzés alkalmával kezdhetjük újra az egészet). Ezután menjünk jobbra le (erre azért van szükség, hogy az ajtón mindig be tudjunk menni, ellenkező esetben azonban orvul bezáródik utánunk), ezt követően menjünk balra, balra, balra, jobbra le, jobbra fel, jobbra, jobbra, itt vegyük a ékköves koronát (jewelled crown), majd menjünk balra, balra, balra, jobbra

le, jobbra fel, jobbra, jobbra le, menjünk a varázsló ajtajához, ami azonnal eltűnik, menjünk a varázslóhoz...

A 6. varázsló kiszabadítása:

A bezárt varázsló felett lévő üveget (large bottle) vegyük fel, menjünk jobbra le, az üst alatt van egy kulcs (door key), menjünk neki az ajtónak, amire kinyílik. Vegyük fel a kulcsot, majd menjünk jobbra le, jobbra fel, jobbra lent, két kis üveg utáni ajtóhoz érve az eltűnik, ekkor vegyük a varázspálcát (magic wand), menjünk balra le, azután ismét balra le, itt vigyázzunk, mert az ajtó alatt víz van, nehogy beleessünk! Ezután menjünk balra le, a varázslóhoz, nyissuk ki az ajtót, szálljunk rá a varázslóra...

A 7. varázsló kiszabadítása:

Ez csak arról a pályáról lehetséges, ahol a 1. számú varázsló volt bezárva. Ha már itt vagyunk, akkor induljunk el balra fel, balra le, itt vegyük fel a gyertyát (fleur de lys), majd menjünk vissza (tehát jobbra fel, ismét jobbra fel, majd középre le, ahhoz a helyhez, ahol az 1. varázsló volt). A gyertyával be tudunk menni az ajtón, itt vegyük a kulcsot (door key), menjünk rá az ajtószerű 'dugó'-ra, ki fog nyílni. Battyogjunk vissza (jobbra), majd menjünk jobbra fel, jobbra le, itt vegyük fel az arany kelyhet (golden chalice), majd menjünk balra fel, balra fel, középre le, és menjünk be az ajtón. Menjünk le középen a nyíláson, majd lent menjünk neki jobbra az ajtónak. Most azonnal, mielőtt továbbmennénk, vegyük fel az üveget (large bottle), majd menjünk le jobbra, menjünk át a nyikorgó faajtón, és végül menjünk neki a varázslónak...

A 8. varázsló kiszabadítása:

Ez a szoba két részre van osztva. Ha fent vagyunk, akkor keressünk egy üveget (large bottle), ezzel kinyílik a 'dugó'. Ezután menjünk le. Ha már lent vagyunk, sétáljunk el jobbra le, balra fel, és vegyük fel a papírtekerccset (scroll). Menjünk vissza (jobbra fel, balra középső ajtón be), csámborogjunk a beszorult varázsló tájékára, majd menjünk neki az ajtónak, ezt követően a varázslónak is...

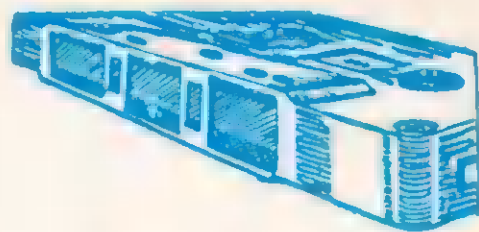
A játék befejezése:

Menjünk jobbra le, jobbra fel, balra fel. Itt egy emelvényt látunk 9 oszloppal. Középen találjuk a legmagasabbat, mellette 4-4 oszlop mindegyikének tetején egy-egy varázsló foglalta a megérdemelt helyét. Egy dolog maradt hátra, hogy mi is elhelyezkedjünk, még üresen maradt oszlop tetején. Ekkor...

Néhány jótanács:

- a 'dugó'-szerű ajtókat, a vékony ajtókat vagy a kulccsal (door key), vagy az üveggel (large bottle) tudjuk kinyitni;
- a 'szörnyet' egy nagy fába vágott fejszével (sharp axe) tudjuk megölni;
- repülni nem tudó, földön mászkáló boszorkányt karddal (strong sword) tudjuk ártalmatlanná tenni;

- az energiánkat az üstnél tudjuk feltölteni;
- ■ nem nyíló nyikorgó faajtókat a címerrel (coat of arms) tudjuk kinyitni;
- Két robbanósszer ■ rendelkezésre a 'szörnyek' totális megsemmisítésére. Az egyik egy száguldó csillagra (shooting star), ■ másik pedig egy pénzszákra emlékeztet, \$ jellel az oldalán. Ha ezeket aktivizáljuk, ügyeljünk arra, hogy ■ velük való robbantáskor egyvonalban legyünk a szörnyel(-ekkel), mert csak így robban(-nak) fel.



■ SORCERY igen izgalmas játék, reméljük, hogy most már könnyűszerrel teljesíthető ■ küldetés.

Amurote ■ Mastertronic + (?&&%?\$\$\$#####???)

Az ENTERPRISE-ra készült SPECTRUM program átiratokat figyelve, már nagyon profi munkákat is fellelhetünk. Ez ■ program is ebbe ■ kategóriába tartozik, ugyanis nem ■ "BAMBA" software ház terméke, ■■ elég ha csak annyit említünk, hogy ■ 128K SPECTRUM hangjával zenél.

Valójában mi is a játék célja?

Az országunkat előzőnlötték a gyilkos méhek, ezeket ki kell itanunk, de lehetőleg úgy, hogy ■ városokban minél kevesebb kárt okozzunk. Ennek a feladatnak a teljesítéséhez egy lépegető terepjáró áll a rendelkezésünkre, melyet helikopterrel juttatnak ■■ megtisztítandó városokba. A gépünk bombákkal ■■ felszerelve, melyekkel ■ méheket kell kipusztítanunk. Ez a feladat ■■ is olyan könnyű, mert a célzás elég nehéz ■ 3D felépítésű tájon. Minden – a rovarokkal történő – érintkezés növeli ■ sérülések mértékét (DAMAGE). A programozó bácsik gondoskodtak arról is, hogy ne unatkozzunk, mivel nem csak a dolgozókkal kell elbánnunk, hanem ■ jó édes mamájukat is fel kell bosszantanunk. Naná, hogy ehhez spéci bombára van szükség! Ilyet (SUPABOMB), valamint

- normál bombát (MORE BOMBS)
- menekítést a meleg helyzetekből (RESCUE)
- javítást (REPAIR)

korlátozott számban – AMIGA készlet tart – ■ <SHIFT> billentyűvel kérhetünk, ezen túl ■ kis kurzort mozgatva választhatunk a fentebb felsorolt opciók között.

Biztos feltűnt már ■ képernyő alján kulmináló három méh-sejt. A középső az aktuális célpontot, a két szélső a hozzánk viszonyított irányát jelöli. A célpontválasztás ■ következőképpen lehetséges:

- Dolgozó <Z> billentyű
- Királynő <X> billentyű
- A már rádión megkért bomba <V> billentyű
- A játéktér színe <C> billentyű

Nos, most már mindenki elmondhatja: profi AMAUROTE játékos (az más kérdés, hogy ezt ki hiszi el nekünk)!!!

Tessék szépen észrevenni ■ szuper grafikát, amelyet a kezdésnél láthatunk, és ez minden újratekésztél idegesítően előjön. Ja, és ha nincs örökéletű verzió, – nekünk is csak Spectrum-ra van –, valószínűleg sok ■ jutunk vele. Formalin ■ kivételnek!




```

; Ha nem egyezik (váltottattunk ■ SCREEN-en)
AAF9 01 FF FF LD BC,FFFF
; öngyilkos lesz
AAFC 11 49 EE LD DE,EE49
AAFF ED 80 LDIR
AB01 21 11 EB LD HL,EB11
; Itt folytatja, ha helyes a SCREEN
AB04 01 4A 00 LD BC,004A
; EB11H-n 4AH mnnyiségű byte-ot XOR-ol
AB07 7A LD A,D
; az ellenőrző összeggel
AB08 AE XOR (HL)
AB09 77 LD (HL),A
AB0A 23 INC HL
AB0B 0B DEC BC
AB0C 78 LD A,B
AB0D B1 INC C
AB0E C2 07 EB JP NZ,EB07
AB11 03 INC BC
; Látszólag értelmetlen rész, ezt ■
AB12 55 LD D,L
; területet módosítja az előző rutin
AB14 21 00 40 LD HL,4000
; Az ellenőrző összeget képző szubrutin
AB17 01 00 1B LD BC,1B00
AB1A AF XOR A
AB1B AE XOR (HL)
AB1C 57 LD D,A
; D-ben az ellenőrző összeg
AB1D 23 INC HL
AB1E 0B DEC BC
AB1F 78 LD A,B
AB20 B1 OR C
AB21 7A LD A,D
AB22 C2 CB E8 JP NZ,E8CB
AB25 C9 RET

```

Ezek után láthatjuk, milyen gonosz volt ez illető. Nézzük végig a programot. Az E8C4H-n található rutin előállít egy számot. Ezt ki-számolhatjuk ■ eredeti SCREEN-ből, de felesleges, elég meg-nézni ■ E8D6H címen levő byte-ot. Az első néhány sort mindjárt megspórolhatjuk. EB01H-n ■ első lényeges rész, ■ XOR-olás. Mivel tudjuk ■ ellenőrző kódot (ha valaki ■ tudná, 22H), cseleken írhatunk egy kis szubrutint, ami elvégzi helyettünk ■ cselekedetet.

Mazochisták ■ kézzel ■ megpróbálhatják. Mivel ■ elején található programrészre nem lesz szükségünk, ide beírhatjuk alkotásunkat. Ehhez szálljunk ki ■ listázásból (STOP vagy az ESC billentyű), majd nyomjuk meg az "M"-et. A módosítás címének kérésére adjuk ■ AAEB-at. Az ENTER lenyomása után kiíródik ■ memóriadump, amiben tudunk módosítani. Szép sorban, az ENTER lenyomása nélkül írjuk be ■ következő számokat:

21 11 ■ 06 4A 7E EE 22 77 23 10 F9 C9

Majd szálljunk ki ■ módosításból (STOP vagy ESC). Ha kilistá-zuk ezt ■ kis programocskát, ■ következőket láthatjuk:

```

AAEB 21 11 AB LD HL,AB11
AAEB 06 4A LD B,4A
AAED 7E LD A,(HL)
AAEE EE 22 XOR 22
AAFO 77 LD (HL),A
AAF1 23 INC HL
AAF2 10 F9 DJNZ AAED
AAF4 C9 RET

```

Futtassuk le ("G" parancs, majd AAEB, ENTER), majd ha rende-■ megkaptuk a "Returned from CALL ■ AAEB" üzenetet, lis-tázzuk ki ■ AB11H címet. Micsoda változás!!

```

AB11 21 77 77 LD HL,7777
AB14 01 A4 38 LD BC,38A4
AB17 CD 47 EB CALL EB47
AB1A 21 00 EE LD HL,EE00
AB1D 01 00 12 LD BC,1200
AB20 CD 47 EB CALL EB47
AB23 21 77 77 LD HL,7777
AB26 01 A4 38 LD BC,38A4
AB29 CD 51 EB CALL EB51
AB2C 21 00 EE LD HL,EE00
AB2F 01 00 12 LD BC,1200
AB32 CD 51 EB CALL EB51
AB35 21 10 A7 LD HL,A710

```

```

; A karakterkészlet kezdőcíme
AB38 22 36 5C LD (5C36),HL
; ■ megfelelő változóba
AB3B 01 5E 2F LD BC,2F5E
AB3E AF XOR A
AB3F ED 42 SBC HL,BC
AB41 31 2F 75 LD SP,752F
AB44 C3 6F ■ JP 006F

```

Tanulságul álljon itt ■ program ■ használt két szubrutint is!

```

AB47 7E LD A,(HL)
AB4E ED 67 RRD
AB4A 23 INC HL
AB4E 0B DEC BC
AB4C 78 LD A,B
AB4D B1 OR C
AB4E 20 F7 JR NZ,AB47
AB50 C9 RET

AB51 7A LD A,D
AB52 AE XOR (HL)
AB53 77 LD (HL),A
AB54 23 INC HL
AB55 ■ DEC BC
AB56 78 LD A,B
AB57 B1 OR C
AB58 20 F7 JR NZ,AB51
AB5A C9 RET

```

Látható, hogy ■ program többi része itt ■ el van rejtve. Ismét módosítani vagyunk kénytelenek.

A módszer hasonló az előző programcska beviteléhez, de itt az AB0F címet kell módosítanunk az alábbi byte-okra:

16 22 21 77 37 01 A4 38
CD 47 AB 21 00 AE 01 00
12 CD 47 AB 21 77 37 01
A4 38 CD 51 AB 21 00 AE
01 00 12 CD 51 AB C9

Ha ezt kilistázzuk, ■ következőket kell látnunk:

```

AB0F 16 22 LD D,22
AB11 21 77 37 LD HL,3777
AB14 01 A4 ■ LD BC,38A4
AB17 CD 47 ■ CALL AB47
AB1A 21 00 AE LD HL,AE00
AB1D 01 00 12 LD BC,1200
AB20 CD 47 ■ CALL AB47
AB23 21 77 37 LD HL,3777
AB26 01 A4 38 LD BC,38A4
AB29 CD 51 ■ CALL AB51
AB2C 21 00 AE LD HL,AE00
AB2F 01 00 12 LD BC,1200
AB32 CD 51 ■ CALL AB51
AB35 C9 RET

```

Ha nem ezt látjuk, akkor vagy valami beleröpült ■ szemünkbe, vagy elírtunk valamit (minden bizonnyal ■ Tisztelt Olvasó). Miután ezzel megvagyunk, futtassuk le ezt is ("G" AB0F, ENTER), és már meg is van a futtatható (átíráható) programunk. Már csak egy akadály ■ hátra, nevezetesen ■ indítási cím. Aki azt hiszi, hogy ilyen már találtunk elegend, téved. Meg kell keresnünk a -sorrendben ■ harmadik - helyes belépési pontot. További tor-turák helyett ■ már ■ tudomására hozzuk azokat ■ fanatikus programozóknak, akik idáig kitartottak (jutalom: 77B2H illetve ■ 4000H eltolásos módszer esetében 37B2H). Hogy ■ honnan származik? Ezt ■ Olvasóra bizzuk (csak annyit segítségül, hogy A710H-2F5EH az pontosan 77B2H, valamint ■ SPECTRUM ROM-ban ■ 6FH címen egy szál utasítás van, ■ pediglen egy JP (HL)).

Miután így sikerült lefegyvereznünk a védelmet, akár rá is térhetnénk ■ program tulajdonképpeni átírására. Sajnos azonban ■ kis újjgyakorlás sok helyet elvett, így ■ tényeg legközelebb-re marad. Addig is, senki ne feledje el kimenteni ■ verejtékes munkával feltöltött programrészeket. Míg mindenki szívszorogva várja következő proféciánkat, el lehet kezdeni ■ ismerkedést ■ programozók stílusával, meg lehet próbálkozni egy ENTERPRISE LOADER készítésével.

Végezetül érdemes összefoglalni az eddigi file-ok adatait.

SCREEN 4000H-ra kell tölteni, 1800H hosszban.
CODE1 7776H-ra kell tölteni, ■ hosszban.
CODE2 E8C4H-ra kell tölteni, 173CH hosszban.

Ha mindent betöltöttünk, el kell ugrani ■ 77B2H címre.

Kellemes bogarázást!

Micro PROLOG

A PERIFÉRIÁK KEZELÉSE

A Spectrum Világ 15. számában már ismerkedtünk a Micro PROLOG file kezelési lehetőségeivel. Azóta levelet kaptunk, amelyben Olvasóink kérték bennünket, hogy alapszinten ismertessük a program periféria kezelési lehetőségeit, talán abból célból, mert esetleg több – ott nem egyértelmű – információra adna magyarázatot.

A Micro PROLOG T1.0 változata csak képernyőt, billentyűzetet, magnetofont és ZX-nyomatót kezel, de

- ZX-nyomató helyett eleve csatlakoztatható minden olyan készülék, a ROM-rutinok ZX-nyomatóként látnak (amelyek pl. BASIC-ben működik a COPY parancs).
- Közlünk egy egyszerű programmodosítást, mely a rendszer betöltése előtti megnyitással lehetővé teszi a #3 logikai készülék szokásos használatát (és javítjuk azt a hibát, melynek eredményeként minden sor második pozíciója ?, ha pl. programot listázunk).
- A PIO reláció segítségével a felhasználó különféle készülékeket programozhat (pl. botkormányt).
- A rendszerben elő van készítve a RS-232 csatorna mindkét irányú kezelése; túl nagy nehézségek árán ez életrekelthető.
- A microdrive periféria használatának megoldása több, de reálisan elvégezhető mennyiségű munkát igényelne (a #3 természetesen irányítható a már említett módosítással is microdrive-ra).

A képernyő és a billentyűzet

A képernyőre rajzolni és írni lehet, továbbá minden, amit a rendszer vagy saját programunk billentyűzetről olvas, megjelenik a képernyőn is (ez nem vonatkozik a billentyűzet PIO relációval való vizsgálatára és a program futása közben beírt, megjelenés előtt puffertöréssel eltüntetett adatokra).

Rajzolni a felső 11 sorba lehet, pontok és vonalak megadásával, a PNT ill. a LNE reláció kiértékelésével. HYBRID üzemmódban a szöveges információk a alsó négy sorba kerülnek (ha csak a BASIC-beli AT-nek megfelelő és a következő sor ill. oszlopindex kiírása meg nem növeli a alsó rész sor-igényét), ekkor tehát a rajz első 20 sora elválik a szövegtől. NORMAL üzemmódban szöveg is a felső 22 sorba írható. Írásra a P és a PP reláció kiértékelése való, bár sok más reláció is ír a képernyőre.

Ha a utolsó szövegsor alá újabb szöveget írunk, akkor a szöveges képernyő-rész tartalma, a BASIC-ben megszokott scroll? Kérdés nélkül, eggyel feljebb lép: a első sor eltűnik. Ilyenkor a közös rajz-szöveg terület (HYBRID üzemmódban a 21-22, NORMAL üzemmódban a 1-22) is mozog. A nyomtatás STOP (<SS+A>) megnyomásával legállítható, majd bármely gomb megnyomásával továbbindítható (ebbe a "bármely"-be nem tartoznak a módváltások). NORMAL üzemmódban képezhetünk szöveges rajzokat a V vezérlő karakter segítségével (mely a BASIC-beli AT-k megvalósító CHR\$22 pontos megfelelője; hasonlóan használható a TAB-t megvalósító CHR\$23 párja, de ne feledjük, hogy ez is két következő karaktert értelmez).

A rendszer aszerint van indításakor NORMAL vagy HYBRID üzemmódban, hogy közvetlenül előtte a képernyő melyik része volt outputra kijelölve. A betöltő program a rendszer indítása előtt ír a képernyő felső részére, így NORMAL állapotban indít; ha külön betöltjük a kódot és pl. RANDOMIZE 30752 beírásával indítjuk, akkor a állapotú képernyővel indul. Indulásakor a "papír" színe sárga, a "határ" fehér, a "tinta" fekete, a képernyő nem villog és nem fényes. A felsorolt attribútumok megváltoztatására különféle relációk adnak lehetőséget. CLS, BORDER, LNE és PNT-nél erre külön argumentumok szolgálnak; P-nél a szövegbe illesztett @P (INK), (PAPER), (FLASH), (BRIGHT), (INVERSE) illetve (OVER) vezérlő

relációk használhatók, utánuk írva a BASIC rendszerből ismert jelentésű karaktereket. Ezek a karakterek általában szintén a segítségével adhatók meg: a kódja 1, a kódja 2 és így tovább a-ig, aminek 31 a kódja. A 0 kódérték körül némi variáció van, ugyanis a kódja 64 (pl. a-1 ir ki, de ugyanez a CHAROF reláció segítségével is ellenőrizhető); a 0 kódú karakternek nincs a segítségével megadható megfelelője (pl. ?((CHAROF 0)(PP X)) egyetlen ?-t ír ki, mintha a egy közönséges karakter kódja volna, éppen a képernyőre író ROM-rutin ?-t ír helyette). Egy karaktersorozatba 0 kódú karakter a STRINGOF reláció segítségével illeszthető, ha előtte CHAROF-fal 0-t adunk kód értékül a változónak. (A 41-95 intervallumon kívüli kódú karakterek előtt álló a hatástalan, magát a-1 kivéve, mert a a-ként lehet szövegkonstansba illeszteni – ha a párban áll, akkor összeolvad a utána következő karakterrel.) Ez a megoldás kissé körülményes, ezért a 207=CFh kódú karaktert a rendszer 0-ként írja ki (?-ként jelenik meg, ha nem vezető karaktert követ, a CAT-billentyűvel a <E-mód 9> – vihető be).

A vezérlő karakterek kezelésére jellemző, hogy ugyanaz a közvetlen hatásuk, mint BASIC-beli megfelelőiknek (a micro-PROLOG is a a rutinjait használja). Ertérés a tovagyűrűző hatásban mutatkozik: a micro-PROLOG általában együtt állítja a ideiglenes attribútumokat a állandókkal, a micro-PROLOG (pl. BASIC-ban PRINT CHR\$18; CHR\$1; "A";PRINT"B") kiír egy villogó A-t és egy már a villogó B-t) ?((P@RGMA)(P B)) mindkét karaktert villogóként írja, sőt, villog utána minden, amit akár a rendszer, akár a program kiír, míg ennek egy olyan reláció – mondjuk CLS – kiértékelése véget nem vet, mely a villogást kifejezetten leállítja). Az állandó és a ideiglenes attribútumok elválasztására csak a PNT és a LNE reláció kínál lehetőséget.

A és PP között a alapvető különbség, hogy az első változatlan formában átadja a kiírandó karakterek sorozatát a képernyőre író ROM-rutinak, míg a második minden olyan szövegkonstansra idézőjel-pár közé zár, mely egyébként nem egy szövegkonstansként értelmeződne. Így lehetővé teszi a vezérlő karakterek rendeltetésszerű használatát, PP pedig a összetett szövegkonstansok tartalmának megjelenítését. PP kiértékelése a képernyőn való sorváltással fejeződik be.

A kódértékük szerint a felhasználói karakterek (UDG) után következő, a legalább A5h=165 kódú karakterek a képernyőre nem írathatók ki. Helyettük általában ? jelenik meg (kódjukat a ROM-rutinok hívása előtt cseréli ki a rendszer a ? kódjára, illetve CAT esetében 0-ra), kivéve a DEF FN-hez tartozó, 206=CCh kódú karaktert, ami egyáltalán nem íródik ki (nem változik a cursor-pozíció a képernyőn), hangjelzést kapunk helyette.

A billentyűzet olvasására elsősorban a R reláció szolgál, mely kiértékelésekor a argumentumában szereplő változónak a billentyűzet-puffer következő elemi kifejezését adja értékül, ill. olvastat, ha a puffer üres, vagy a kifejezés hiányos (pl. végzárójel v. idézőjel-pár záró eleme hiányzik). K-mód nem állítható, egyébként a reláció a BASIC-ből ismert módon értelmezi a karaktereket. Kivétel a 127 kódú c karakter, mely más rendszerekben törést jelent, olyan mintha nem is lenne (a lyukszalagos kor-szakban hibajavításra szolgált: a lyukszalagnas szalagon a hét értékes lyukhely mindegyikét kilyukasztva lehetett törölni egy hibás lyukkombinációt). Az R reláció is nemlétezőnek tekinti, ha önmagában áll, nem egy szövegkonstans része. (Értékelteessük ki ?((R X)(PP X))-t egyszer (A c B)-t, másodszor ("a" "c" "B")-t beírva a számára: A visszairt lista a első esetben (A B), a másodikban ("c" B) lesz.)

Az input billentyűzetről való megadásakor a <ENTER> megnyomásáig a beírt szöveg a BASIC-ben megszokott módon szerkeszthető a és <DELETE> segítségével. Ha túl próbálunk lépni a beírt karaktersorozaton, akkor a rendszer hangjelzést ad ("kiír egy DEF karaktert"). A szerkesztéshez rendelke-

zésre álló puffer mérete 8 képernyő-sor, ■■■■ 256 karakter (ebből egyet a cursor foglal el). A parancs futása közben is beírható 16 karakter (a szerkesztőket ■ beleértve); ezeket ■ megkezdett input-puffer feldolgozása után (a kiírása után) veszi figyelembe ■ rendszer.

File-ok

A file-ok olyan, általában ■ számítógép perifériáin elhelyezkedő adatszerkezetek, melyek kezelésénél nagymértékben eltérünk ■ adathordozó sajátosságaitól. Az ■ előny, hogy ilyen – logikáinak is nevezett, magas – szinten szemlélhetjük, mozgathatjuk adatainkat, általában bőségesen kárpótol azért ■ hátrányért, hogy ■■■■ használhatjuk ki ■ konkrét adathordozó lehetőségeit (ha nem így van, akkor lehet szükség ún. fizikai szintű perifériakezelésre).

A micro-PROLOG T1.0 csak ■■■■ szövegfile-okat kezel, melyeket csak írhatunk és olvashatunk (file-jaiban nincsenek logikai rekordok, ■ esetleges blokkolást ■ programok elől eltakarja, nem lehet egy-egy részletet, módosítva, eredeti helyére visszairni). Néhány file ■ rendszerben eleve adott:

CON: író műveleteknél ■ képernyőre írás, olvasó műveleteknél ■ billentyűzetről olvasás file-ja. A képernyőkezelésnél ismertett P, PP és R reláció, definíciója szerint, egyenértékű ■ CON:ra hivatkozó W, WRITE illetve READ relációval.

LST: ■ ZX nyomtató file-ja.

RDR: ■ RS232 input file-ja (pontosabban annak kezdeménye, mert érdemi része hiányzik, adatátvitelt nem végez).

PUN: ■ RS232 output file-ja (pontosabban annak szintén csak kezdeménye, ■ előbb említett ok miatt).

Ezek ■ file-ok mindig használhatók, ■ rájuk hivatkozva kiértékelte OPEN és CREATE relációk mindig sikeresek (ha ■■■■ csak input-ra alkalmas fájlra adunk CREATE-t vagy csak outputra alkalmasra OPEN-t) és alapjában véve hatástalanok.

A felsoroltakon kívül ■ program még egy felhasználói file-tal képes dolgozni, melynek adathordozója ■ ■■■■ vagy MIC jelű csatlakozónál bekötött magnetofon. E file neve minden, más cél- ■ még nem foglalt, legfeljebb nyolc karakter hosszú szövegonstans lehet (így akár ■ üres is, amit mára nem ■ lehet lefoglalni). Nincs elvi akadálya a microdrive-kezelés megoldásának, de egy ilyen továbbfejlesztés meghaladná ■ munka keretelt (az irodalomból nyilvánvaló, hogy ■ gyártó ezt ■ fejlesztést végrehajtotta, de nincs tudomásunk a fejlettebb program hazai előfordulásáról).

A magnetofon

Fizikai szintű kezelését a ROM rutinjai végzik – írásuk a 4C2h-n induló rutin működik, olvasásuk ■■■■ induló eleje helyett egy saját változatot fut, majd ■ vezérlés ■ eredeti rutin megfelelő oimére kerül (az eltérés logikailag érdektelenek). A blokkok hossz- ■■■■ bevezető bajtjuk értéke FBh. Minden blokkban ■ első 255 byte lehet értékes, ■ ■■■■ mindig bináris 0, ezt követi ■ file neve – szükség esetén szóközzel kiegészítve – nyolc karakteren, majd ■ blokk karakteres ábrázolású sorszáma (01 ■ első; ■■ után 00, majd ismét 01 következik). Csak az utolsó blokkban nincs kihasználva ■ első 255 byte: ■■■■ blokkot ■ még hátralévő értékes karakterek vezetik be, majd utánuk ■ file végét jelző 26=1Ah kódú karakter áll (ha ezzel nem merül ki ■ lehetséges 255 byte, a folytatás akkor is érdektelen – az előzőleg kezelt blokk vége van ■ file-ban; kedvezőtlen esetben ■ file végjele egyedül is elfoglalhat egy blokkot).

■ felhasználói file-ok létrehozását egy CREATE reláció kiértékelése ■ vezeti be, ■■■■ követhetik olyan relációk, amelyek írnak bele, végül egy CLOSE reláció teszi ■ file-t teljessé. Ha ■ CLOSE előtt újabb CREATE következik ugyanarra ■ file-ra, akkor készítése előlről kezdődik; ha CLOSE előtt OPEN jönne, akkor ■ logikailag mindenekelőtt egy CLOSE-t hajt végre. Ha egy író reláció eredménye már nem tér ■ file soronlevő blokkjába, vagy CLOSE-t értékel ki ■ rendszer, akkor képernyőre íródik a file neve és blokkjának sorszáma, majd megindul ■ szalagraírás. Ilyenkor ■

gép kezelőjének felvételre kell kapcsolnia ■ magnetofont, majd – ha nem jön azonnal következő blokk – célszerű leállítania.

A felhasználói file-ok olvasását egy OPEN reláció kiértékelése vezeti be, ezt követhetik olyan relációk, melyek olvasnak belőle, végül egy CLOSE reláció jelzi, hogy a file-ra nincs tovább szükség. Ha CLOSE előtt újabb OPEN következik, előlről indul ■ file olvasása, ha pedig CREATE előzi meg ■ OPEN-t, akkor meg-

kezdődik a file ismételt létrehozása (tovább nem olvasható). Ha egy file-t nevének pontos megadásával olvasunk, akkor ■ rendszer ■ OPEN reláció kiértékelésekor kiírja nevét 01 sor-számmal, majd igyekszik beolvasni a file első blokkját ■ csak ennek sikeres megtörténte után engedi tovább ■ programot. Ha ■ következő reláció-kiértékelések végigolvasták ■ bentlövő blokk értékes részét és nem jutottak ■ ■ file végét jelző 26 (1Ah) kódú karakterig, akkor ■ rendszer ismét kiírja ■ file nevét, mellé ■ következő sorszámot, majd igyekszik beolvasni ■ megfelelő blokkot ■ csak ennek sikeres megtörténte után engedi tovább ■ programot. (A fizikai olvasás ■ blokk utolsó karakterének átadásakor zajlik le, nem akkor, mikor ■ következő első karakterére szükség van!)

A rendszer megengedi, hogy egy file ■■■■ üres string (") – vagy ezzel egyenértékűen szóköz – legyen, de nem csak ■ ilyen file-okat olvashatjuk úgy, hogy névként üres string-et adunk meg ■ rájuk vonatkozó relációkban. Ha a blokkok 01-től sorban követik egymást fizikai olvasáskor, akkor mindössze annyi ■ eltérés ilyenkor ■ kezelésben, hogy OPEN-nél csak ■ ■■ sorszám jelenik meg, ■ file-név helye üresen marad. Következő olvasáskor ■ sorszám eggyel nő, ■ file-név ■ először olvasott név lesz (így csak akkor nem változik, ha üres string nevű file-ból olvasunk). Ilyenkor azonban ■ rendszer nem ellenőrzi, hogy ■ olvas- ■■■■ blokk sorszáma megfelelő-e, hagyja feldolgozni, majd ■ most már számára ismert nevű file következő blokkját várja. Ha ■■■■ jó helyen áll ■ szalag, akkor ebben ■ esetben általában hibát okoz a nem megfelelő blokk feldolgozása. A micro-PROLOG T1.0-nak ez ■ tulajdonsága lehetővé teszi, hogy ■ üres string nevű file-ok blokkjait tetszőleges sorrendben, egy blokkra akár többször is sort kerítve dolgoztassuk fel, csak utolsóként olyan blokkot olvassunk, mely végjelet tartalmaz (ennek neve más is lehet).

Általában ajántható, hogy adjunk a file-oknak valódi nevet és csak akkor olvassuk őket nevük helyett üres string-et adva, ha biztosan tudjuk, hogy elsőként ■ 01 sorszámú blokkot fogja ■ rendszer megkapni (pl. szalag elején állunk, vagy előtte egy file-t végigolvastunk – ■ magnetofonokba épített számláló állása ritkán elegendő biztosíték).

Ha a beolvasott blokk megfelelő, akkor ■ előtte kiírt név és sorszám után megjelenik ■ BLOCK OK felirat és ■ kiírási hely ugyanennek ■ sornak ■ eleje lesz. Így sorozatos olvasásnál ugyan ide kerül ■ következő blokk neve és sorszáma, letörölve ■ előző BLOCK OK feliratot. ■■■■ kiértékelése sort vált ■ képernyőn, így a folyamat minimális szakmá sor-leptetéssel jár. Ez ■ helytakarékos megoldás zavart okozhat olyankor, ha nem sorozatban olvassuk ■ blokkokat, hanem közben használjuk ■ képernyőt más célra is. Hogy ■ billentyűzött input nem üres sorban jelenik meg ■ képernyőn, ■ csak némi többlet-figyelmet igényel, de hogy egy esetleges hibauzenet számát jobbról kiegészíti ■ blokkorszám második jegye, ■ komoly talány lehet.

Ha ■ beolvasott blokk ■■■■ az, amit ■ rendszer várt, akkor kiírja ■ nevét és sorszámát – ugyanoda, ahova egyébként BLOCK OK-t – , majd olvassa ■ következő blokkot. Ha olvasási hiba van (más a bevezető byte értéke, rövidebb ■ blokk vagy nem egyezik ■ hosszanti paritás), akkor ugyanott READ ERROR felirat jelenik meg és szintén ■ következő blokk olvasása következik (a kezelőnek megfelelő helyre kell tekernie ■ szalagot).

A nyomtató

A rendszer nyomtató-kezelése ZX-nyomtatót tételez fel, melyre ugyanaz és ugyanúgy írható, mint képernyőre (eltérőként olyan természetes különbségektől, hogy ■ nyomtató nem tudja visszahúzni ■ papírt és nem színes). Legegyszerűbben úgy nyomtathatunk, hogy ■ TO billentyű (<SS+F>) megnyomásával bekapcsoljuk ■ "másolat" (hardcopy) funkciót. Ennek ■ ■■ eredménye,

hogy minden, amit a rendszer ■ **CON:** file-ra ír, kikerül ■ **LST:** file-ra is. E funkció **TO** ismételt megnyomásával állítható le. Fontos megjegyezni, hogy ■ **CON:**-ről érkező input nem kerül át **LST:**-ra, így a készülő lista nem dokumentálja ■ végzett munkát, nem "igazi hardcopy".

A nyomtató használatának szokásos módja ■ **LST:** file-ra való írás. Erre elsősorban ■ **W** és ■ **WRITE** reláció kiértékelése szolgál, bár más relációk is írhatnak fájlba, így írhatunk nyomtatóra is. Rajz nyomtatására ■ rendszer nem ad lehetőséget, ■ képernyőre készült rajzok sem másoltathatók segítségével nyomtatóra (bár ■ utóbbi gépi kódú megoldása minden konkrét, "rajzolni tudó" nyomtatóra egyszerű feladat).

Ha olyan nyomtatóval dolgozunk, melyet ■ **ROM**-rutinok ZX-nomtatóként érzékelnek, akkor csupán a vezérlő karakterekkel kell óvatosan bánnunk; minden ugyanúgy használható, mint **BASIC**-ben.

Ha másképp szeretnénk nyomtatni – pl. **RS232** felhasználásával vagy **microdrive**-ra irányítva ■ nyomtatandókat –, akkor beleutközünk ■ rendszer egy viszonylag egyszerűen áthidalható korszakába: ■ **micro-PROLOG T1.0** mindig a ■3. logikai csatorna rutinjaival kezelteti ■ képernyőt, ■ billentyűzetet és ■ nyomtatót, ahelyett hogy az utóbbi esetben ■ 3. logikai csatornához rendelt rutinnal dolgozna. Hogy ■■ tegye, ■ következő módosítások elegendőek:

929Ch-től 3 bájtra írjuk ■ következőt: **CDh, 6Fh, 97h;**

976Fh-től 17 bájtra pedig ■ következőt: **2Ah, 4Fh, 5Ch, FDh, CBh, 01h, 4Eh, C8h, D6h, EDh, 5Bh, 1Ch, 5Ch, 2Bh, 19h, D1h, C8h.**

Ha a gépi kódú program indítása előtt ■3-t a megfelelő fizikai csatornához rendeljük, akkor a nyomtatás eredménye a hozzárendelt készülékre kerül (**microdrive** esetén néhány dologra ügyelni kell: egyszer ■ program előtt nincs hely két **microdrive**-csatorna számára, ezért ha a programot is onnan töltjük be, akkor ■3-t a betöltés után nyissuk meg; másszor nincs ■ rendszerben mód ■3 lezárására, ezért némi fölösleges írással gondoskodjunk arról, hogy ■ értékes információ vége is ■ adathordozóra kerüljön, ahonnan **BASIC**-ban ■■ segítségével vehető elő; harmadszor hiba esetén – pl. ha megtelik ■ kazetta – ■ vezérlés kikerülhet a interpreterből és nincs garancia arra, hogy munkánk eredménye nemvész el, ■ **RANDOMIZE** ■■ valószínűleg segít).

RS232 esetén, ■■ nyomtatót vezérelni akarjuk, használjuk ■ **B** csatornát.

Bármilyen nyomtatónk van, találkozunk néhány könnyen javítható hibával. Egy rossz ugrás miatt bizonyos karakterek másképp kerülnek nyomtatóra, mint képernyőre – elsősorban ■ **CAT** ■■ használható 0 kódú karakter kiírására (ami ■ vezérlést nehezíti). Javítása: írjunk **91EB**-ra **94h**-t.

LISTP feltételezi, hogy ■ kiíró rutinok megőrzik ■ A regiszter értékét. A **CON**-ra és a felhasználói fájlokra író rutin valóban megőrzi, ■ többi azonban nem. A nem működő **PUN**: érdektelen, míg életre nem keltik, **LST**: esetében megfelel ■ következő javítás:

735Dh-től 2 bájtra írjuk a következőt: ■■ **97h**:

9780h-től 6 bájtra pedig ■ következőt: **F5h, CDh, 76h, 91h, F1h, C8h.**

Ezáltal ■ eredeti, regisztertartalom-rontó rutin-hívást betesszük egy mentés-visszatöltés pár közé és így ■ nyomtatott listákon is jó lesz a sorok eleje.

KILLED UNTIL DEAD

LEVEL III. - CASES FOR THE CUNNING

1. pálya

(The Case of the Mutilated Moose)

gyilkos: Peter

áldozat: Sydney

hely: Patio

fegyver: gun

ok: He ran over your brother

Break In

Sydney - -

Peter - Bina Crossby (3.)

Claudia - -

Agatha - Bare Knuckle

Boxing

Mike - Qua Seraisera

2. pálya

(The Mystery of the Leaping Fish)

gyilkos: Claudia

áldozat: Peter

hely: hall

fegyver: gun

ok: Peter stole your "fish" plot

Break In

Sydney - Traris McGee

Peter - Soff soled shoes

Claudia - Sherlock

Holmes

Agatha - Murp the Surf

Mike - Cricket

3. pálya

(Paint by Numbers)

gyilkos: Claudia

áldozat: Peter

hely: hall

fegyver: gun

ok: Peter ruined your reputation

Break In

Sydney - Vincenzo

Agatha - -

Peter - Mary Tyler Moore

Mike - -

Claudia - Blackjack

4. pálya

(Practical Pastimes)

gyilkos: Mike

áldozat: Agatha

hely: library

fegyver: gun

ok: She pulled too many practical joker

Break In

Sydney - Dr. Antuirete

Louis

Peter - Sir Alic Guinness

Claudia - Baker Street

Agatha - Beekeeping

Mike - Maigret

5. pálya

(A Stich in Time)

gyilkos: Mike

áldozat: Claudia

hely: Mike's room

fegyver: knife

ok: Claudia squeezed you out of the deal

Break In

Sydney - A circus

elephant

Peter - Jimmy Hotta

Claudia - Chicago

Agatha - Yankee

Mike - Billy the Kid

Desolator

CHEAT ÖTLET

A játék **MULTILOAD**-os, azaz az egyes szintek csak az előzők teljesítése után tölthetők be. Ezt átverhetjük, ha egy előző pálya fejece után egy következő pálya fő-kódját gepeljük be. Meglátjuk, így a további szintek is játszhatók!

HISOFT 'C' COMPILER

A programszerkesztő használata

A Spectrum C rendszerének használatát a programszerkesztő ismertetésével folytatjuk, majd ezután térünk rá magának a nyelvnek az ismertetésére. Mint említettük a szerkesztőbe a **EDIT** majd a **ENTER** billentyűk megnyomásával léphetünk be. A szerkesztő minden számmal kezdődő sort beszerkeszt a pufferébe, hasonlóan a BASIC-hez. A sorszám nem lesz része a program-sornak, csak a szerkesztéshez szükséges. A szerkesztőnek egy-karaktéres parancsai vannak, melyek a következők:

L<sorszám1>, <sorszám2> – automatikus sorszámadás. Az első szám az első sornak a száma, míg a második szám a növekmény. A szerkesztő addig adja a megfelelő sorszámkat, míg be a vizunk egy sort, amelyik csak az **<EDIT>** kódját (kis téglalap) tartalmazza.

L<sorszám1>, <sorszám2> – listázás az első számtól a második számig.

K<sorszám> – beállítja, hogy hány soronként történjen a listázás.

W<sorszám1>, <sorszám2> – ugyanaz, mint az L, csak a nyomtatóra listáz.

V – Kijrja az aktuális elválasztójelet és az utóljára használt **<sorszám1>**, **<sorszám2>**, **<param1>**, illetve **<param2>** értékeket, továbbá a szövegfile kezdő és végcímét.

S, **<param1>** – a parancsok paraméterei közt általában a vessző áll. Ha valamiért nem megfelelő, akkor átcsereíthatjuk a **<param1>** sztring első karakterére. Ezt nem célszerű szököznek, vagy **ENTER**-nek választani!

C – Visszatérés a fordítóba.

B – Kilépés a BASIC-be. Újraindítás a **RANDOMIZE USR 25200** parancsral!

N<sorszám1>, <sorszám2> – a sorok átszámozása. Az első megadott szám a első sor száma, míg a második a növekmény.

F<sorszám1>, <sorszám2>, <param1>, <param2> – megadott sorok közt (beleértve a határokat) keresi a **<param1>** sztringet, s ha megtalálta, áttér szerkesztő módba (lásd később). Lehetőségünk van a soron belül további előfordulást keresni, vagy a megtaláltat helyettesíteni a **<param2>** sztringgel. A sztringeket nem szabad idézőjelek közé tenni!

P<sorszám1>, <sorszám2>, <param1> – puffer megadott sorait kiírja a szalagra **<param1>** néven.

G, **<param1>** – beolvassa a **<param1>** nevű file tartalmát a pufferbe.

Ez utóbbi két parancs esetén, ha a sztring második karaktere kettőspont, akkor a előtte álló szám a microdrive sorszáma, a név pedig a harmadik karaktertől kezdődik.

Utoljára hagytuk a **E<sorszám1>** szerkesztési parancs ismertetését. Ennek segítségével a létező, **<sorszám1>** sorszámú sort tudjuk megszerkeszteni. Ehhez az alábbi alparancsokat használhatjuk:

<Kurzor jobbra> – egy karaktert átmásol a régi sorból a újba.
<Kurzor balra> – visszarajka a átmásolt karaktert a régi sorba.
<ENTER> – a szerkesztés befejezése, az új a bekerül a puffer-be.

Q – a szerkesztés vége, az eredeti sor megmarad.

I – a eredeti sor szerkesztését elolról kezd.

Z – törli a kurzor után álló karaktereket.

K – a kurzor alatti karakterrel együtt töröl a sor végéig.

I – beszúrás. Villogó a kurzor jelenik meg. A felesleges karaktereket a **<DELETE>** gombbal törölhetjük. A beszúrás befejezhetjük az **<ENTER>** megnyomásával.

I – a összes maradék karaktert átmásolja a sor végére áll, s áttér beszúrási módba.

C – helyettesítési mód. A kurzor a lesz, a a bevitt karakterek átírják az eredeti sorban levő karaktereket. Kilépni az **<ENTER>** megnyomásával tudunk.

F<param1>, <param2> – befejezi a sor szerkesztését, kicseréli a régít és megkeresi a következő sort, ahol a F után beírt sztring megtalálható.

S – a F parancsban megadott sztringre cseréli a megtalált sztringet, majd tovább keres.

A szerkesztő parancsai és a E parancs alparancsai úgy működnek, hogyha valamelyik paraméterüket (ezek a **<sorszám>** és **<param>** értékek) a adjuk meg, akkor a utóljára megadott értékekkel dolgozik. Ezeket lehet a V parancsral lekérdezni. Tekintettel arra, hogy a C program képes önmaga felülírására, minden programfuttatás előtt célszerű a 10 sornál hosszabb

programokat elmenteni! Javításnál nem kell mindig, csak akkor, ha már nem emlékszünk, mit is javítottunk ki.

A C nyelv általános tulajdonságai

Egy C program változó és típusdeklarációk valamint függvények halmaza. A függvényeken belül már nincs lehetőség további függvények deklarálására, míg a függvényen belül használhatunk további változódeklarációkat. Ennek következtében a változókat külső és belső változóknak hívjuk, attól függően, hogy az összes függvényen kívül, vagy valamelyiken belül deklaráltuk. (Más nyelvek esetén a globális-lokális elnevezéspár használatos.) A belső változók még a deklarációjuktól függően lehetnek statikusak vagy sem. A statikus változó értéke a függvényből való kilépéskor is megmarad és a függvény újabb hívásakor ezzel a értékkel számolhatunk tovább. Valamennyi használt változót a első használata előtt a deklarálni kell.

A C nyelv maga lehetővé teszi a **extern** és **register** típusú változók használatát is. Ez utóbbi azt fejezi ki, hogy a változót magát, ha ez lehetséges, a processzor regiszterében valósítsuk meg. Erre a Spectrum esetében nincs lehetőség, a regiszterei másra már foglaltak. Az **extern** típus azzal van összefüggésben, hogy a C nyelven írt program egyes darabjait külön file-ban is tárolhatjuk, külön fordíthatjuk. Ilyenkor jelezni kell a file elején, hogy melyek azok a függvények és változók, amelyek deklarációját egy másik file tartalmazza. Erre nekünk nincs lehetőségünk, mert a fordító mindig a forrásfile-ból és egyetlen menetben állítja elő a futtatható kódot.

A C erősen típusos nyelv: minden változónak van típusa, ezt a program szövegében expliciten meg is kell adni. A C aránylag kevés típust ismer, ezek a következők:

| C elnevezés | Jelentés | Tárolási hossz |
|-------------|--------------|------------------|
| 1. char | karakter | 1 byte |
| 2. int | egész szám | 2 byte |
| 3. short | egész szám | 2 byte |
| 4. long | egész szám | 2 byte |
| 5. float | valós szám | nem implementált |
| 6. double | dupla pontos | nem implementált |

Általában a C megvalósításokban a **int**, **float** és **long** típusok nem azonosak, de ebben a reprezentációban igen. Mint a bevezetésben már említettük, a valós számokat nem kezeli a rendszer. A 2-4. típusok elé tehetünk egy **unsigned** módosító szócskát, ilyenkor a illető változók nem tárolhatnak negatív számot.

A változó-deklarációk alakja igen egyszerű:

[**static**] <típus megnevezése> <változó lista> ;

Például

```
int i,j;
char beolv, kiir, c;
short alfa, beta;
static unsigned ciklusvalt;
```

mind érvényes deklarációk. Ha a **static** jelzőt nem tesszük ki, akkor a változó a lesz statikus, amit néha úgyis ki szoktunk fejezni, hogy a változó automatikus. (Természetesen a alaptípusokon kívül vannak ún. típusképző operációk – pl. tombók – amelyek segítségével további típusokat lehet definiálni.)

Az alaptípusokon – elsősorban a számokon – a megszokottnál lényegesen több műveletet lehet végezni, a azok jelölése is meglepő. Ezek a műveletek (és relációk) a következők:

| C jelölés | elnevezés |
|-----------|--------------------------------|
| 1. ! | logikai tagadás |
| 2. ~ | bitenkénti komplementer-képzés |
| 3. + | összeadás |
| 4. - | kivonás, ellentett képzés |
| 5. * | szorzás, mutatóképzés |
| 6. / | osztás |
| 7. % | maradék képzés |
| 8. < | balra tolás |
| 9. > | jobbra tolás |
| 10. <= | kisebb |
| 11. >= | nagyobb |
| 12. <= | kisebb egyenlő |
| 13. >= | nagyobb egyenlő |
| 14. == | egyenlő |
| 15. != | nem egyenlő |
| 16. & | bitenkénti és |
| 17. | bitenkénti vagy |
| 18. && | bitenkénti kizáró vagy |
| 19. | logikai vagy |
| 20. 0 | sorozatós kiértékelés |
| 21. ? : 2 | feltételes kifejezés |
| 22. ++ | növelés |
| 23. -- | csökkentés |

| | | | |
|-----|-----------------|---|--|
| 24. | = | 1 | értékkadás |
| 25. | += | 1 | értékkadás összeadással |
| 26. | -= | 1 | értékkadás kivonással |
| 27. | *= | 1 | értékkadás szorzással |
| 28. | /= | 1 | értékkadás osztással |
| 29. | %= | 1 | értékkadás maradékképzéssel |
| 30. | >>= | 1 | értékkadás jobbróltozással |
| 31. | <<= | 1 | értékkadás balróltozással |
| 32. | &= | 1 | értékkadás bitenkénti és-sel |
| 33. | = | 1 | értékkadás bitenkénti vagy-gyal |
| 34. | ^= | 1 | értékkadás bitenkénti kizáró vagy-gyal |
| 35. | sizeof | | változó tárolási hossza |
| 36. | cast(< típus >) | | konvertálás |

Az egyoperandusú műveletek prioritása a legmagasabb (nem számítva a zárójeleket). Ezek a következők: `!, *, &, -, |, ~, ++, --, sizeof, ...`. Ezek az operátorok jobbról balra kötnek. Az összes kétoperandusú művelet balról jobbra köt. Ezek prioritását a táblázatban soroltuk fel. Ezek után következnek prioritásban az értékkadó műveletek, amelyek mindegyike jobbról balra köt. Végül legalacsonyabb prioritású a vessző művelet, s így mindig hajtódik végre utoljára. A vessző balról jobbra csoportosít.

A C nyelvben az értékkadás mint önálló utasítás nem jelenik meg, hanem speciális műveletként szerepel. Az `x=2` értékkadás eredményül 2 ad, s mellesleg az érték az `x` változóba is belemásolódik. Pl. az `y += (x=2)` művelet korrektt. Eredményül az `y+2` értéket kapjuk, s mellékhatásként az `x` változóba kerül 2 érték.

Hasonlóan furcsa a vessző művelet, bár a HISOFT-ban nem implementálták. Mégis megemlítjük, mert a C programokban igen gyakran használják. Pusztán annyit jelent, hogy a vesszővel elválasztott kifejezéseket sorban ki kell értékelni, s magának a kifejezésnek az értéke az utoljára kiértékelt kifejezés lesz. Tekintsük például az `i = (j = 2, j + 1)` kifejezést! Először természetesen a zárójelen belüli kifejezés értékelődik ki. Ennek hatására `j` értéke 2 lesz, majd kiértékeldődik `j + 1` kifejezés, melynek értéke így 3 lesz. Ezért `i` 3 értéket kapja, s magának az egész kifejezésnek az értéke is 3 lesz!

Következő furcsaság a `++` típusú értékkadások. A C nyelv készítői úgy találták, hogy igen gyakoriak az `X=X+1` típusú értékkadások, s ezek gyors végrehajtása érdekében bevezették a `++` értékkadást. `X++=Y` az `X=X+Y` értékkadás rövidítése. Ha tehát az `X` változót 1-gyel akarjuk növelni, akkor a leggyorsabban azt az `X++=1` paranccsal hajthatjuk végre. Hasonlóan kell értelmezni az összes többi értékkadó műveletet.

Az 1-gyel való csökkentést és növelést annyira lényegesnek találták, hogy külön csökkentő és növelő műveleti jelet iktattak a nyelvbe, ezek a `++` és `--` jelek. Ezeket csak változó elé vagy után lehet írni. Ha elé írjuk, akkor a változó értékét először kell módosítani, majd utána használni. Ha utána írjuk, akkor előbb használjuk a változó értékét, csak utána módosítja a program. Például az `i = (x = 2, y = 3, x + (++y))` művelet hatására `i` értéke 6 lesz! (Vajon miért?)

Utoljára maradt a feltételes értékkadás. Ez egy háromargumentumú művelet: `<feltétel> ? <kifejezés> : <kifejezés>`. A feltételes kifejezés értéke a követő első kifejezés, ha a feltétel igaz, különben a második. Például az `x > y ? x : y` kifejezés értéke mindig az `x` és `y` számok közül a nagyobbik.

C programok felépítése

Mint említettük, a C program változódeklarációk és függvénydefiníciók sorozata. A program futása a `main` nevű függvény meghívásával kezdődik, ez tekinthető tulajdonképpen a főprogramnak. Egy függvénydefiníció az alábbi alakú:

```
[<extern> <típus>] <név> [<paraméterlista>]
<változó-deklarációs lista 1>
{ <változó-deklarációs lista 2>
  <utasítási lista> }
```

A `<név>` a függvény nevét adja meg, ezzel a névvel kell a függvényre majd hivatkozni. A paraméterlista a függvény formális paramétereit adja meg, ezeket és csak ezeket a változó-deklarációs lista 1-ben deklarálni kell. Ezeket a változókat csak és kizárólag a függvény belsejében használhatjuk. A kapcsos zárójelek közti részt szokás függvénytörzsnek hívni. Ez ugyancsak tartalmazhat egy deklarációs listát, ezek lesznek a függvény belső változói. Végül a függvénytörzsben kell megadni a végrehajtandó utasítások sorozatát. A függvény nevét egy típusazonosító előzheti meg, megadja, hogy a függvény milyen típusú értéket tér vissza. Ha elmarad, akkor az mindig `int` típust jelent. Annak jelzésére, hogy a függvény visszaadott értéke érdektelen, szokás a `void` típust használni. A `void` típus valójában egész típust jelent, de a függvény neve elé írásával egyértelműen jelölhetjük, hogy nincs felhasználható visszaadott érték. Az `extern` használatára még későbbiekben visszatérünk.

A továbbiakban felsoroljuk az összes C utasítás formáját. Ügyeljünk egyes utasítások végén látható pontosvesszőre (;) használatuk kötelező!

1. Összetett utasítás. Az utasítás alakja:

```
{ <változó-deklaráció lista> <utasítási lista> }
```

alakú, hasonló a függvénytörzssel. Az itt szereplő változók csak az összetett utasításon belül léteznek, onnan való kilépéskor értékük nem használható fel.

2. A `<kifejezés>;` szintén utasítás. Hatására a `<kifejezés>` kiértékeldődik, s értéke rendelkezésre áll. Ha a kifejezésben értékkadások is szerepeltek, akkor a kifejezés kiértékelése egyben az értékkadások végrehajtását is jelenti.

3. Függvényből visszatérni kétféleképpen lehet. Vagy a függvény törzsének utolsó utasítását is végrehajtjuk, vagy kiadjuk a `return` utasítást. Annak alakja kétféle:

```
return ; vagy return <kifejezés> ;
```

Ez utóbbi esetben a függvény értéke a `<kifejezés>` lesz, míg az előző esetben nem definiált. Az első (tehát a `<kifejezés>` nélküli) esetet akkor használjuk, ha nem akarunk a függvényvel értéket visszaadni.

4. `goto` utasítás. Bár a C strukturált nyelv, lehetőség van címke használatára a programban, s megcímkezett utasításra való közvetlen vezérlésátadásra. Erre szolgál a `goto` utasítások címkével való ellátása, illetve a `goto` utasítás. Ezek alakja a következő:

```
goto <címke> illetve <címke> : <utasítás>
```

A címke tetszőleges, másra még nem használt azonosító lehet. A `goto` utasítás végrehajtása azt eredményezi, hogy a vezérlés a `<címke>`-vel megjelölt utasítással folytatódik. A címke csak éppen végrehajtás alatt álló függvényben lehet.

A feltételes vezérlésátadásra

```
if ( <kifejezés> ) <utasítás> illetve
```

```
if | <kifejezés> ) <utasítás 1> <utasítás 2>
```

utasítások szolgálnak. A kifejezés kiértékelése után a vezérlés az `<utasítás 1>` feldolgozásával folytatódik - feltéve, hogy a kifejezés értéke igaz volt, pontosabban nullától különböző. Ha a kifejezés értéke hamis, azaz nulla, akkor az első esetben a következő utasításra kerül a vezérlés, míg a második esetben az `<utasítás 2>` kerül végrehajtásra. A C tartalmaz egy többirányú elágaztatást is lehetővé tevő utasítást. Ennek alakja a következő:

```
switch ( <kifejezés> ) <utasítás>
```

míg a `<utasítás>`-ban az alábbi utasítások fordulhatnak elő:

```
case <állandó-kifejezés> : <utasítás> illetve default :
<utasítás>
```

A `switch` utasítás végrehajtása a `<kifejezés>` kiértékelésével kezdődik. Ezután a program megkeresi az utasításban az első olyan `case`-t, amelyhez tartozó állandó kifejezés értéke megegyezik a `<kifejezés>` éppen aktuális értékével. Az ezt a követő első utasítást folytatódik a program. Ha ilyen `case` nincs, akkor a default-ra kerül a vezérlés. Ha a default nem szerepel a `switch`-en belül, akkor a követő első utasításra kerül a vezérlés. Szemben a C nyelv definíciójával, a HISOFT C-ben a `switch` utasításon belül már nem definiálhatunk semmit.

5. `while` utasítás. A C háromféle ciklusutasítást ismer. Ezek a következők:

```
while ( <kifejezés> ) <utasítás>
do <utasítás> <kifejezés>;
for ( <kifejezés 1>; <kifejezés 2>; <kifejezés 3> )
<utasítás>
```

A két `while`-t tartalmazó struktúra hasonló. Az `<utasítás>` rész újra és újra végrehajtódik egészen addig, amíg a `<kifejezés>` igaz, azaz nem nulla. Amikor a `<kifejezés>` értéke először lesz nulla, a ciklus lejár. A `do`-val kezdődő ciklus annyiban tér el, hogy az `<utasítás>` legalább egyszer végrehajtódik.

A `for` utasítás egyenértékű a következő ciklussal:

```
<kifejezés 1>;
while <kifejezés 2> {
  <utasítás>
  <kifejezés 3>
}
```

Ennek megfelelően az első kifejezés inicializálja a ciklust, a második ellenőrzi a ciklus lejárát, míg a harmadik a ciklus egy-egy végrehajtása után módosítja a ciklusváltozó értékét. Például a BASIC-ből jól ismert `FOR I=1 TO N STEP 1` ciklust a C-ben a következőképpen lehet kifejezni: `for (i=1; i++ <= n)`.

6. Egyéb utasítás. Ebbe a kategóriába összesen három utasítás tartozik. Ezek közül a legegyszerűbb az üres utasítás, azaz a `;`. Felesleges pontosvessző üres utasításként viselkedik. A

break utasítás befejezi az utasítást tartalmazó legbelső **while**, **do, for** vagy **switch** utasítást, s az azt követő első utasításra kerül a vezérlés. A **continue** utasítás befejezi a ciklusmag végrehajtását, s a ciklus folytatásának ellenőrzésére kerül vissza a vezérlés. Ez a kétféle **do** ciklusra egyaránt vonatkozik.

Előre definiált függvények

Befejeztük a C nyelv ismertetésének első részét. A továbbiakban a típusdeklarációkat, az előre definiált függvényeket és fordítási opciókat ismertetjük. Jelenlegi ismereteink azonban még nem elegendőek programok írására, ugyanis a C-ben az adat ki/beviteli műveletek mind előre megírt függvények (hasonlóan a PASCAL-hoz), ezért legalább néhány eljárást ismeretünk, hogy tudjunk már most programokat készíteni.

Adatbeolvasás

A billentyűzetről történő adatbeolvasás formája a következő:

```
scanf(<formátum-sztring>, <argument>...);
```

A formátum-sztring a beolvasandó értékek formátumát tartalmazza, míg az argumentum-lista a beolvasandó értékek helyét adja meg, vagyis mindegyik egy-egy mutató. A formátum-sztringben egész értékeket a **%d** vagy a **%10d** formában kell jelezni. A **%** jel vezeti be a vezérlő sorozatot, míg a **"d"** decimális beolvasási formátumot jelzi. A köztük lévő szám a maximális beolvasandó jegyek számát jelenti. Bármilyen decimális jel véget vet a beolvasásnak. Például **%x** és **%y** egész számokat a következőképpen lehet beolvasatni:

```
scanf("%d%d",&x,&y);
```

Kiírás a képernyőre

A képernyőre való kiírás ugyancsak formátum-sztring segítségével történik. Ennek alakja:

```
printf(<formátum-sztring>, <argument>...);
```

Argumentumokként most magukat az értékeket megadni, nem pedig címüket. A **<formátum-sztringben>** található nem vezérlő karakterek egy egyben megjelennek a képernyőn. A kiírás vezérlésére a következő karaktersorozatokat is lehet még használni:

| C jelölés | hatás |
|-----------|----------------------------|
| \n | Új sor |
| \t | vízszintes tabulálás |
| \b | visszalépés |
| \r | kocsi-vissza |
| \f | lapdobás (képernyő törlés) |
| \a | backslash |
| \\ | apoztróf |

Például a **printf("\n\nEredmények: %5d %5d",x,y)** hatására a képernyő törlődik, a második sor elején megjelenik az "Eredmények:" felirat, majd azt követően 5-5 helyiértéken ábrázolva az **x** és **y** számok értékét, közte két szóközzel.

Most már kellő ismerettel rendelkezünk ahhoz, hogy elég bonyolult és összetett programokat írjunk C nyelven. Bemutatóként egy egyszerű példa.

Az alábbi program az első 1000 természetes szám között található prímszámokat írja ki a képernyőre:

```
int oszthato(i,j)
int i,j;
{
    if(i%j == 0) return -1; else return 0;
}
main()
{
    int i,j,flag;
```

```
printf("%10c%d\n",CLR,2);
for(i = 2; i < 1000; i += 2) {
    flag = 0;
    for(j = 2; j <= i > 1; (j = 2)?j++:j += 2))
        if(flag == oszthato(i,j)) break;
    if(! flag)
        printf("%d\n",i);
}
```

Jelen ismertetőnkkel azzal fejezzük be, hogy ismertetjük, hogyan kell egy C programot megírni és futtatni: Töltsük a fordítót, majd bejelentkezés után nyomjuk meg az **<EDIT>** **<ENTER>** billentyűket. Ennek hatására szerkesztő üzemmódba kerülünk. Adjuk ki a **10,10** szerkesztő parancsot, majd ahogy a program sorban adja a számokat írjuk be egyes programsorokat. Vigyázat: a C alapszavakat csak kis betűkkel írhatjuk be! Amikor valamennyit beírtuk, akkor a beszüntési üzemmódból úgy tudunk kilépni, hogy egy olyan sort írunk be, amelyik egyedül az **<EDIT>** billentyű jelet (kis téglalap) tartalmazza. A fentiek elvégzése után nyomjuk meg még a **<C>** billentyűt is. Ennek hatására a szerkesztőből visszatérünk a fordítóba. A **#INCLUDE <ENTER>** után a fordító lefordítja a szerkesztő pufferében lévő programot. Ezután már csak a **<SYMBOL SHIFT-I>** (vagy **AT**) billentyűt kell megnyomnunk. Ha nem vétettünk hibát, akkor a fordítótól üzenetet kapjuk, hogy a **<Y>** gomb megnyomására elindul a program.

Elképzelhető, hogy mind a forrásnyelvi, mind a lefordított programot szalagra vagy mikrodrive-ra szeretnénk átmásolni. Ekkor a következőképpen kell eljárunk: ha a forrásnyelvi szöveget akarjuk elmenteni, akkor a szerkesztőből ki kell adnunk egy **P10,150**, "ozanavem" parancsot. Vigyázzunk: a sorszámoknak általunk megírt program első és utolsó sorának a számával kell megegyeznie! Ha a lefordított programot akarjuk elmenteni, akkor a program szövegének első sorába a **<filenév>** alakú vezérlő sort kell elhelyezni. A file mentésére a **<SYMBOL SHIFT-I>** billentyű megnyomásakor kerül sor. A lefordított és elmentett program visszatöltése után a **USR** BASIC parancssal indítható el.

Rendben, nézzük, hogyan működik a program! Rögtön látszik, hogy két függvényből áll, ezek a **oszthato** és a **main**. A **main** főprogram, ezért ennek nem lehet paramétere. Az **oszthato** nevű függvény kétváltozós és egész értéket ad vissza. Ez 0, ha a maradék nélkül osztható j-vel, különben -1. A **oszthato** függvénynek három belső változója van, valamennyi egész. Ezek a **i**, **j** és a **flag** nevű változók.

A **main** kiírja az első prímszámot, ami a 2. (Ha valaki nem tudná: egy szám akkor prímszám, ha az 1-en és önmagán kívül maradékok nélkül egyetlen pozitív számmal osztható...) Ezután következik egy ciklus, ami hárommal indul és 999-ig tart. A ciklusmag minden egyes lefutása végén végrehajtódik a **i += 2** kifejezés kiértékelése, aminek következményeként i mindig kettővel nő. A belső ciklus ellenőrzi, hogy van-e az i-nek osztója. Ez a ciklus **j = 2** értékkel indul, s egészen **i/2**-ig tart. Az **i/2** értéket trükkösen **"i > 1"**-nek írtuk be. A ciklusváltozót **(j = 2)?j++:j += 2** kifejezés segítségével módosítjuk. Ha j egyenlő kettővel, akkor j egygyel nő a **j++** kifejezés hatására, minden más esetben kettővel, mert ilyenkor a **j += 2** rész kerül kiértékelésre. A ciklusváltozó ilyen használata meggyorsítja a ciklust, hiszen a páros számok – kivéve a 2-t – kimaradnak. A ciklusmag egyetlen feltételes utasításból áll. A feltétel kiértékelésének egyik melírhathatása, hogy a **flag** értéke mindig beállítódik. Ha a feltétel igaz (**< > 0**), találtunk osztót, akkor a **break** miatt kilépünk a ciklusból. Végül ha a **flag** jelzi, hogy a számnak nincs osztója, akkor a második **printf** kiírja a számot. Ha még valaki itt van: írjunk olyan programot, amelyik kiírja egy szám prímtényezőinek felbonthatását. Használjunk minél bonyolultabb és trükkösebb értékadásokat!!!

LED STORM • GO!

Amikor begyűjtött pontszámunk leszámolása történik, nyomjuk meg egyidejűleg a **<CAPS SHIFT>** és a **<SPACE>** billentyűket (BREAK), ekkor a képernyő kerete zöld színűre vált, a játék futása pedig felfüggesztődik. Most nyomjuk meg az aktuális tűz-gombot az újraindításhoz, lám 300.000 ponttal rendelkezünk.

TASK FORCE • CCS/Premier Software

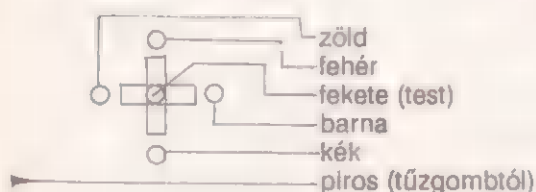
Amikor definiálnunk kell a billentyűzetet, gépeljük be sorban: C, H, E, A, T, majd ezt követően pedig definiáljuk nekünk megfelelő billentyűket. Az induláskor végtelen élettél fogunk rendelkezni.

Numerikus billentyűzet

Köztudott, hogy a Spectrum billentyűzetén – még ■ 128+2, +3 billentyűzetén is – nehéz számokat, adatokat, ■ numerikus számítások során alkalmazott szimbólumokat (tizedespont, szorzásjel stb.) bevinni. Egy speciális célprogram (pl. bérelszámolás, bruttó-sítás stb.) esetén ez a probléma fokozottabban jelentkezhet. A 128K géptípusokhoz igaz forgalomba került egy numerikus külső billentyűzet, amely a KEYPAD csatlakozón keresztül illeszthető, ám úgy gondoljuk 48K-s gépből valamivel több található az országban, az ő problémájukat kellene elsősorban megoldani.

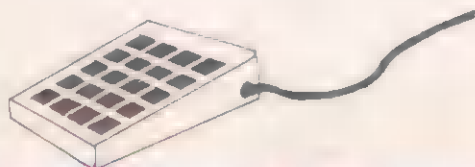
A megoldás kulcsa egy billentyűzetre programozható joystick interface – ilyenből sokféle létezik, még hazai forgalmazásban is –, egy lerobbant joystick, valamint a **külső numerikus billentyűzet**, melynek megépítésére most fogunk ötleteket adni.

Az első lépés ■ joystick szétszerelése. Célszerű, ha egy már egyébként használhatatlan joystick-et használunk fel erre ■ célra. A joystick kis nyáklapján meg fogjuk találni az oda csatlakoztatott kábelvégződést. Ez gyakran úgy néz ki, hogy minden ér más színű. Ezermester üzletekben kapható többeres szalagkábel (laposkábel), vegyünk ebből egy kb. 70 cm-es darabot. Ebből vágjunk le egy 20 cm-es részt, fejtünk le belőle 6 különböző színű eret, majd pucoljuk meg ■ végeket, és forrasszuk az egyik oldali végződéseket a csatlakozásokhoz:



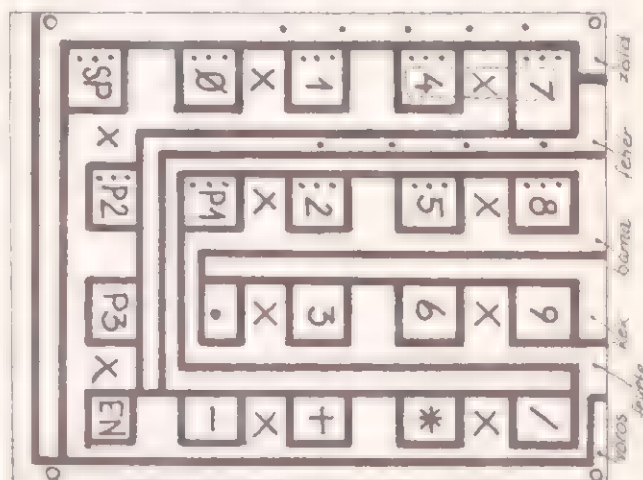
A szalagkábel másik végén is pucoljuk meg a vezetékvégeket, majd forrasszuk fel rá egy 6 pólusú tuchel dugót. A 6 pólusú tuchel dugót lengő aljzattal együtt szerezzük be, az aljzatot majd a numerikus billentyűzethez fogjuk felszerelni. Feltétlenül jegyezzük meg azt is, hogy melyik kábelszín melyik pólus-hoz tartozik.

Ezt követően szereznünk kell egy kisebb méretű műanyag dobozt. A doboz méretét persze ■ beépítés alapvetően befolyásolja, ám nem akarunk konkrét adatokat megadni (a x b x c), ■■ menet közben úgy is ki fog derülni. Lombfűrészszel vágjuk ki ■ billentyűk helyét, 10 x 10 mm-es nyílásokat, ugyanakkor vegyünk egy kb. 6-8 mm vastagságú gumilapot, ebből pedig vágjunk ki 8 x ■ mm méretű hasábokat (ezek lesznek ■ billentyűk).



A műanyag doboz felső lapjára (amelyen ■ nyílásokat kivágtuk) alulról vágjunk be egy megfelelő méretű gumilapot (célszerű rossze kerékpár belsőből), majd pillanatragasztóval ■ nyílásoknak megfelelően ragasszuk fel a billentyűket ■ vékony gumilapra. Amikor minden billentyű ■ helyére került, célszerű várni néhány órát pihenni, hogy ■ ragasztó kötés szilárdsága megfelelő legyen, ezután ■ számokat és a jeleket felvihetjük a billentyűk felületére. Ennek megoldását már Önökre bizzuk, akár égetéssel, akár festéssel.

Amíg a ragasztó megköt, nekiláthatunk a nyáklap elkészítésének. A kapcsolást egy – ■ vékony gumilappal megegyező nagyságú nyák-lapon fogjuk megtervezni, ez házilag is könnyen megoldható. A nyák felületén ■ vastag sávokat maratjuk ki, ■ szín-jelölések az általunk elkészített billentyűzetre vonatkoznak, lehet bármilyen más vezeték-szín kombináció.



A számozott négyzetek szélén és a csíkokon készítsünk Ø 1 mm-es furatokat, a szerkezet szívét képező egyenirányító diódák részére, ugyanis ezekkel lehet megoldani, hogy a joystick-kel ellentétben, ■ joystick átlós irányának megfelelő kettős kapcsolást megkapjuk. Ide már ■ leggyengébb egyenirányítók is megfelelnek, kb. 7,- Ft/db, és 35 db-ra lesz szükségünk. A nyákhoz forrasszuk ■ szalagkábel másik részét (kb. fél méter), a színeknek megfelelően, a másik végére pedig szereljük fel ■ tuchel lengő aljzatot. A diódákat ■ kis furatokon keresztül ■ nyák síma oldala felől fel-dugva forrasszuk be. A fő irányoknak (2,4,6,8) megfelelő kapcsolásokat dióda nélkül, azokból lecsípett darabokkal oldhatjuk meg. A fő irányok betartása fontos a joystick betanításához! A diódákat úgy forrasszuk fel, hogy ■ azokon látható "csík" jelzés mindig ■ kiinduló mezőhöz (pl. 7,9) essen közelebb. Az irányába mutasson. Részletesebb rajz helyett ■ leírjuk, hogy mely mezőtől mely színhez csatlakozzon ■ dióda.

| | | |
|---|---------|--------------|
| ■ | – fehér | dióda nélkül |
| 4 | – zöld | dióda nélkül |
| 2 | – kék | dióda nélkül |
| ■ | – barna | dióda nélkül |
| 5 | – piros | dióda nélkül |
| 7 | – fehér | diódával |

| | | |
|----|---------|----------|
| 7 | – zöld | diódával |
| 9 | – fehér | diódával |
| 9 | – barna | diódával |
| 1 | – zöld | diódával |
| 1 | – kék | diódával |
| 3 | – kék | diódával |
| 3 | – barna | diódával |
| 0 | – zöld | diódával |
| 0 | – fehér | diódával |
| 0 | – piros | diódával |
| / | – kék | diódával |
| / | – barna | diódával |
| / | – piros | diódával |
| * | – fehér | diódával |
| * | – barna | diódával |
| ■ | – piros | diódával |
| + | – zöld | diódával |
| + | – kék | diódával |
| + | – piros | diódával |
| – | – zöld | diódával |
| – | – piros | diódával |
| SP | – fehér | diódával |
| SP | – piros | diódával |
| EN | – barna | diódával |
| EN | – piros | diódával |
| P1 | – kék | diódával |
| P1 | – piros | diódával |
| P2 | – fehér | diódával |
| P2 | – kék | diódával |
| P3 | – zöld | diódával |
| P3 | – barna | diódával |
| . | – zöld | diódával |
| . | – barna | diódával |
| . | – piros | diódával |

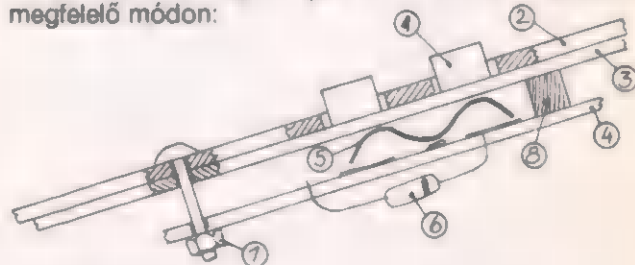
Ahol SP = SPACE, EN = ENTER, ■, P1, P2 és P3 tetszőleges billentyűk.

Amikor ez is kész lett, vágjunk le rugalmas rézlemez-
ből kb. 2 mm-es csíkokat, egyenként kb. 20 mm
hosszút, majd hajlítsuk meg ezeket ■ ábrának meg-
felelő alakra.



Forrasszuk ezeket a csíkokat a nyákrájon X-szel je-
lölt, tehát ■ közös (fekete) érintkezőkre. Attól függő-
en, hogy ■ hajlítás milyen magasra sikerült, ragasz-
szunk a gumilap aljára hálószerűen megfelelő ma-
gasságú gumicsíkokat, ez fogja megadni ■ billentyű-
zet érzékenységet, és meggátolja, hogy másik kap-
csoló is bekapcsoljon.

Most már a billentyűket belülről felnyomhatjuk ■ mű-
anyag dobozon kivágott nyílásokba, a nyákot 4 db.
M3-as csavarral rögzíthetjük a dobozon, ■ vázlatnak
megfelelő módon:



- | | |
|------------------------|-----------------------|
| 1 – billentyű | 5 – rugalmas lemez |
| 2 – műanyag doboz | 6 – dióda |
| 3 – gumilap | 7 – csavar |
| ■ – nyomtatott áramkör | ■ – távtartó gumicsík |

Amennyiben ■ összeszerelés rendben megtörtént,
következhet ■ főpróba! A tanítható, programozható
joystick interface-t tanítsuk be úgy, mintha a joystick-
kel tennénk, ám most ■ numerikus billentyűzet billen-
tyűi segítségével. A számok, műveleti jelek, ■
tizedespon, a SPACE és ■ ENTER mellett rendel-
kezésre álló 3 billentyűt ■ célprogram sajátossággal
szerint választhatjuk ki, tetszés szerint.

Reméljük, hogy hasznos információval láttuk ■ mind-
azokat, akik egy kicsit is szeretnek otthonukban bar-
kácsolni, és gépüket gyakran használják numerikus
feladatokra. A felhasználáshoz sok sikert kívánunk!

KILLED UNTIL DEAD

LEVEL IV. SUPERSLEUTH

1. pálya

(Last Laff)

gyilkos: Claudia

áldozat: Sydney

hely: Agatha's room

fegyver: bomb

ok: Sidney said you were

Cousy writer

Break In

Sydney - 4

Peter - 2

Claudia - -

Agatha - Phillis Dorothy

Mike - -

2. pálya

(Motherly Love)

gyilkos: Agatha

áldozat: Sydney

hely: patio

fegyver: knife

ok: You wanted Mike's
inheritance

3. pálya

(Rhymes and crimes)

gyilkos: Claudia

áldozat: Agatha

hely: library

fegyver: gun

ok: Jealousy

4. pálya

(The Scales of Justice)

gyilkos: Claudia

áldozat: Peter

hely: patio

fegyver: poison

ok: He weighs less than
the rest

Break In

Sydney - Beekeeper

Agatha - Rammers

Claudia - Danger Man

Mike - E.A.Poe

Peter - -

Vége!

A billentyűzet figyelése

A SpV. 15. számában ■ billentyűzet figyelésének BASIC-ből megvalósítható módszerére szemléltettünk egy példát. Sajnos elkövettük egy hibát: a *Beta Basic* segédprogrammal nem mindenki rendelkezik. Sok le-
vélíró is javasolta: ne tegyük feltétlenül szükségessé ■ *Beta Basic* betöltését, ha a módszer az alap BASIC utasításkészletével is szemléltethető. Ezért most ■ ru-
tint egy kicsit átirítottuk.

```
10 REM *** Bill.figyeles ***
20 DIM b$(5): FOR i=16 TO 175
STEP 32: PLOT 0,1: DRAW 255,0: N
EXT i
25 PLOT 0,16: DRAW 0,153: DRAW
255,0: DRAW 0,-153
30 PRINT AT 2,5;"A billentyuk
figyelese"
40 PRINT AT 5,1;"1-5";AT 6,1;6
3486;AT 9,1;"Q-T";AT 10,1;64510;
AT 13,1;"A-G";AT 14,1;65022;AT 1
7,1;"CS-V";AT 18,1;65278
50 PRINT AT 5,28;"0-6";AT 6,26
;61438;AT 9,28;"P-Y";AT 10,26;57
342;AT 13,26;"ENT-H";AT 14,26;49
150;AT 17,27;"SP-B";AT 18,26;327
66
60 PLOT 128,144: DRAW 0,-128:
PLOT 50,144: DRAW 0,-128: PLOT 2
03,144: DRAW 0,-128
70 LET a=IN 61438: PRINT AT 5,
18;a: GO SUB 100: PRINT AT 6,8;b
$: LET a=IN 64510: PRINT AT 9,10
;a: GO SUB 100: PRINT AT 10,8;b$
: LET a=IN 65022: PRINT AT 13,10
;a: GO SUB 100: PRINT AT 14,8;b$
: LET a=65278: PRINT AT 17,10;a:
GO SUB 100: PRINT AT 18,8;b$
80 LET a=IN 61438: PRINT AT 5,
18;a: GO SUB 100: PRINT AT 6,18;
b$: LET a=IN 57342: PRINT AT 9,1
8;a: GO SUB 100: PRINT AT 10,18;
b$: LET a=IN 49150: PRINT AT 13,
18;a: GO SUB 100: PRINT AT 14,18
;b$: LET a=IN 32766: PRINT AT 17
,18;a: GO SUB 100: PRINT AT 18,1
8;b$
90 PRINT AT 20,0;"Aktualis bil
lentyu: ";CHR$(PEEK 23560);"
.
95 POKE 23560,32
100 LET b$="000000"
110 LET x=a-160
120 IF x/16<1 THEN LET b$(5)="1
.
130 IF x/16>=1 THEN LET x=x-16
140 IF x/8<1 THEN LET b$(4)="1"
150 IF x/8>=1 THEN LET x=x-8
160 IF x/4<1 THEN LET b$(3)="1"
170 IF x/4>=1 THEN LET x=x-4
```

```
180 IF x/2<1 THEN LET b$(2)="1"
190 IF x/2>=1 THEN LET x=x-2
200 IF x<1 THEN LET b$(1)="1"
210 RETURN
```

Ellipszis kompozíció

Ez az egyszerű demonstrációs program azt mutatja be, hogy különböző koordinátájú ellipszisek segítségével hogyan tudunk 3 dimenziós hatást keltetni:

```
1 CLEAR
2 INK 2
5 LET x=0: LET y=80
10 FOR n=0 TO 2*PI STEP PI/180
15 PLOT 128+x*SIN n,87+y*COS n
20 NEXT n
25 LET x=x+10: LET y=y-10
30 IF y=-10 THEN STOP
40 GO TO 10
```

Kristály kompozíció

A trigonometrikus függvények és a SPECTRUM rajzoló utasításainak együttes felhasználásával szép, gyémántcsiszolathoz hasonló ábrát kapunk eredményül:

```
10 CLEAR
20 DIM a(12): DIM b(12)
30 FOR n=1 TO 12
40 LET k=n/6*PI
50 LET a(n)=128+80*SIN k: LET
b(n)=88+80*COS k
60 PLOT a(n),b(n)
70 NEXT n
80 FOR n=1 TO 12
90 FOR m=1 TO 12
100 LET ox=a(m)-a(n)
110 LET oy=b(m)-b(n)
120 PLOT a(n),b(n): DRAW ox,oy
130 NEXT m: NEXT n
```

Plusz egy grafika

Ez egy igazi ABS-trakt alkotás, ugyanis az ABS függvény alkalmazásával értük el ezt ■ meghökkentően érdekes hatást!

```
10 REM ABS-trakt
20 FOR k=0 TO 60 STEP 10
30 FOR x=0 TO 128
40 LET y=30-ABS (x-32)*SIN (x*
PI/64)
50 PLOT x,y+k: PLOT y+k,x
60 PLOT 255-x,y+k: PLOT 255-(y
+k),x
70 NEXT x
80 NEXT k
```

SHANGHAI WARRIORS • Players

Amikor feliratkozunk a HIGH SCORE táblára, a nevünk helyett gépeljük be: OUTLAND. Ha most elindítunk egy új játékot, rendelkezésünkre fog állni egy bomba, amelyet minden időben aktivizálhatunk.

SKATEBALL • Electronic Arts

A címkepernyőn gépeljük be: TIXY, ekkor a gép készségesen végigmutatja nekünk mind a 26 szintet, majd végtelen élettel ajándékoz meg bennünket.

Az amatőr programozók döntő többsége csupán a BASIC nyelvet ismeri és használja, holott szívesen beillesztene egy-két gépi kódú rutint is programjaiba olyan feladatok elvégzésére, amelyek BASIC nyelven csak igen dögögösen, túlságosan lassan, vagy egyáltalán nem oldhatók meg. Többek között ehhez kívánunk segítséget nyújtani ■ **Gépi ■ tanfolyam** sorozat előző fejezetei, amelyek a SPECTRUM-ban is alkalmazott Z-80 mikroprocesszor utasításkészletét ismertették több-kevesebb részletességgel. Ezek ■ utasítások képezik a gépi kódú programozás alapelemeit – mondhatnánk építőköveit – amelyekből ■ program felépíthető. A BASIC utasításokhoz szokott programozó azonban ezeket nem tudja ■ megszokott programozási gondolatmenetbe beillesztetni, így – hacsak nem kap segítséget – szinte nehezebben igazodik el közöttük, mint a BASIC nyelvet ■ ismerő kezdő.

A segítség legjobb és legegyszerűbb módja néhány mintaprogram ■ bemutatása, ■ alkalmazott gondolatmenet levezetésével. A következőkben ismertetésre kerülő programok ill. rutinok főleg ezt ■ célt szolgálnak, de olyan gyakorlati feladatok megoldására irányulnak, amelyek amatőr BASIC programokban is sokszor előfordulnak, ■ kidolgozás viszont olyan, hogy ezek ■ rutinok megíró amatőr programokba is beilleszthetők.

Bevezetőül mindenesetre néhány megfontolásra érdemes jó tanács:

1. Tanuljuk meg valamelyik ASSEMBLER/DISASSEMBLER program kezelését, és lehetőleg rendszeresen azt használjuk. Nálunk legelterjedtebb ■ GENS-3, MONS-3 páros, amelyek nagy előnye, hogy ■ memória tetszőleges területére tolhatók be (a mintaprogramok is ■ készültek, de ez ■ zárja ki más monitor-assembler program használatát). Ha mindkettőt betöltjük, akkor az assembler program azonnal ellenőrizhető és futtatható.
2. Legyen kéznél ■ ■ utasításkészlet összefoglaló jegyzéke (mit fogad el ■ mikroprocesszor?) és részletes ismertetője, amelyeket bizony eleinte szinte minden utasítás beírása előtt segítségül ■ hívni.
3. Az assembler programba beírt utasítássorozatot (forráskódot) célszerű futtatás előtt kimenteni, amit minden ■ program ■ teszt.
4. Ne feledkezzünk meg ■ RAMTOP áthelyezéséről CLEAR utasítással, ■ gépi kódú programunk alá, különösen ha ■ memória legfelső részét ■ használni akarjuk. A ROM program ugyanis közvetlenül ■ RAMTOP alatti rekeszeket használja veremterületként, aminek felülírása kellemetlen következményekkel jár.
5. Folyamatábra készítése nemcsak ajánlatos, de összetett program esetén ■ áttekinthetőség csak így biztosítható.

A bemutatásra kerülő mintaprogramok főként demonstrációs célt szolgálnak, ■ legegyszerűbb utasításokból összeállítva. Minden más szempont – pl. tömörség, működési sebesség, kis helyszükséglet – háttérbe került.

Kezdjük ■ gyakorlást egy olyan rutinnal, amely egy rendezetlen számsor tagjait nagyság szerinti sorrendbe rendezi. Egyszerűség kedvéért ■ számsor legfeljebb ■ tagból álljon, és az egyes számértékek is csak 0...255 közötti pozitív egész számok lehetnek (így ut ■ csak egy byte hosszúságú adatokat ■ kezelni).

■ rendezés gondolatmenete a következő:

A számsor első tagját egymás után összehasonlítjuk ■ ■ mögötte állóval, ■ ■ ha valahol egy nála kisebb értéket találunk, ■ két számot felcseréljük. Csere után folytatjuk a műveletet, de már ahhoz az új értékhez hasonlítunk, amit csere útján ■ első helyre beírtunk. A sor végére érve bizonyos, hogy ■ számsor elején annak legkisebb értékű tagja áll. Ezt megismételve ■ második, harmadik stb. taggal, ■ rendezés teljessé válik. A programot ■ memóriában bárhol elhelyezhetjük. Ha nem használunk nyomtatót, akkor kényelmes megoldás ■ nyomtató 23296. címen kezdődő, 256 byte hosszú tárterületének felhasználása, ahol ■ programot semmi nem zavarja, és még ■ RAMTOP-ot sem kell áthelyezni.

Számrendezés

| | | | |
|-----|--------|------|-------------|
| 10 | TAGOK | EQU | 23296 |
| 20 | CIM | EQU | 23298 |
| 30 | CIMTAR | EQU | 23300 |
| 40 | CIKLUS | EQU | 23302 |
| 50 | | ORG | 23310 |
| 60 | | LD | HL,(CIM) |
| 70 | | LD | A,(TAGOK) |
| 80 | | DEC | A |
| 90 | | LD | B,A |
| 100 | C1 | LD | A,B |
| 110 | | LD | (CIKLUS),A |
| 120 | | LD | A,(HL) |
| 130 | | PUSH | HL |
| 140 | C2 | INC | HL |
| 150 | | CP | (HL) |
| 160 | | CALL | NC,CSERE |
| 170 | | DJNZ | C2 |
| 180 | | POP | HL |
| 190 | | INC | HL |
| 200 | | LD | A,(CIKLUS) |
| 210 | | LD | B,A |
| 220 | | DJNZ | C1 |
| 230 | | RET | |
| 240 | CSERE | LD | C,(HL) |
| 250 | | LD | (HL),A |
| 260 | | LD | (CIMTAR),HL |
| 270 | | POP | DE |
| 280 | | POP | HL |
| 290 | | PUSH | HL |
| 300 | | PUSH | DE |
| 310 | | LD | (HL),C |
| 320 | | LD | HL,(CIMTAR) |
| 330 | | LD | A,C |
| 340 | | RET | |

Az egyes lépések magyarázata:

A 10-50 sorok csupán ún. assembler direktívák, amelyek ■ programozást könnyítik. Egyszerűbb ugyanis címkéket beírni mint számértékeket, sőt esélyünk van arra, hogy tévedés esetén az assembler program erre figyelmeztet. Ezekben:

- 10 - a számsor tagjainak számát tároló rekesz címe;
 - 20 - a számsor kezdőcímét tartalmazó rekeszpár;
- Az előbbi adatokat nem rögzítjük ■ programban, hanem esetenként kívülről adjuk meg (BASIC-ből POKE utasítással), hogy a programunk univerzálisan használható legyen;
- 30 - itt ■ mindenkor soron következő szám címét helyeztük el (2 byte);
 - 40 - ez a ciklusváltozó (még hátralévő tagok száma) tárolási helye;
 - 50 - A program kezdőcíme;
 - 60 - Lehívjuk ■ számsor kezdőcímét;
 - 70 - és a tagok számát (n);
 - 80 - ■ csak n-1 lépést kell tennünk.
 - 90 - Ezt azért töltjük át B-be, mert ■ most következő ciklusban ■ B értéket használjuk, ciklusváltozóként.
 - 100 - ■ C1 külső ciklus kezdete, ■ B-ben hordozott ciklusváltozó minden visszatéréskor ■ memória adott rekeszében kívánjuk elraktározni, de oda csak az A regiszterből tudunk értéket betölteni (110)
 - 120 - Mivel itt HL mindig arra ■ címre mutat (és ügyelünk arra, hogy ■ ciklusból való visszatérés után is így legyen), ahol ■ soron következő bázisszám található, ■ ott lévő értéket ■ A regiszterbe töltjük;
 - 130 - ■ címet pedig kimenjük ■ verembe. (A verembe való adattárolás mindig nagyobb körütekintést igényel, mint ■ memóriarekeszben való tárolás, mivel onnan ■ adatok csakis egymás után, fordított sorrendben vehetők ki.)
 - 140 - A belső, C2 ciklus kezdete, amelyben ■ bázisszámot rendre összehasonlítjuk ■ mögötte állókkal. Először ■ következő tag címére lépünk (140), és ■ ott lévő értéket összehasonlítjuk ■ A-ban lévő bázisszámmal (150).

- 160 - Ha \square \square \square érték \square bázisszámnál nem nagyobb, vagyis nem volt átvitel (amit \square C jelzőbit mutat), akkor meghívjuk a CSERE szubrutint. (Vegyük észre, hogy \square program azonos értékek esetén is cserél, de ez \square eredményt nem befolyásolja).
- 170 - B értéke csökken, és amíg 0 értékét \square nem éri, vissza \square ciklus kezdetére. Fontos tudni, hogy \square B regisztert közben nem használtuk, tehát abban még a ciklusváltozó értéke van.
- 180 - A külső ciklust \square következő számmal folytatjuk, amelynek címét \square veremből kivett érték növelésével kapjuk (egy lépés előre).
- 200 - Visszatérés előtt le hívjuk és B-be töltjük a ciklusszámot (210), és értékét csökkentve visszatérünk \square ciklus elejére, ha még van hátralevő tagunk (220). Egyébként vége (230). A CSERE szubrutinba lépéskor a HL regiszterpár az éppen vizsgált szám címét tartalmazza, míg \square bázisszám címe a címtárban, értéke viszont \square A regiszterben van. Ezeket kell egymással felcserélni.
- 240 - Átmenetileg C-be töltjük a vizsgált szám értékét.
- 250 - helyére beírjuk \square bázisszámot;
- 260 - és megőrizzük azt a címet is, ahol \square csere végett megálltunk. Az aktuális kezdőcímet, ahová C értékét be kell töltenünk, \square veremben tároljuk, tehát onnét kell kivenni. Igen ám, de most egy szubrutinban vagyunk, és \square ide lépéskor a ROM program a verem tetején helyezte el a visszatérési címet, alatta helyezkednek el az oda korábban bevitt értékek. Ezért:
- 270 - Kivesszük \square felül lévő adatot bármely használaton kívüli regiszterpárba,
- 280 - majd utána \square keresett címet is, és visszaállítjuk az eredeti állapotot (290, 300), persze ügyelve \square fordított sorrendre.
- 310 - Az aktuális kezdőcíme beírjuk \square új értéket.
- \square - és a programot ott folytatjuk, ahol megszakítottuk,
- \square - de már az új bázisszámmal.

Következő programunk célkitűzése legyen egy **rendezett számsor véletlenszerű, alapos összekeverése**. Ez az igény nem csak \square amatőr játékokban gyakori, hanem \square számítástechnika egyéb ágaiban is. Mi azért maradunk meg \square kezdő amatőr szintjén, és **ennek megfelelően** válasszuk ki \square követendő eljárást is.

lémet vezessük \square azt \square korlátozást, hogy a számsor csak egy byte-os egységekből áll, és tagjainak száma sem több 255-nél. Elhelyezünk egy \square tagú számsort a memória egymást követő részekében, és egy alkalmas üres területet feltöltünk egy ugyan-csak n számú, INT RND n tagokból álló véletlen számsorral, amit BASIC-ből könnyen és gyorsan lehet generálni. (A helypazarlást tekintsük bocsánatos bűnnek). Ezek után \square keverés annyiból áll, hogy végiglépve \square számsoron, valamint az RND számsoron is, \square utóbbi helyén álló véletlenszám azt adja meg, hogy \square számsor adott tagját melyikkel (a sor elejétől számítva hányadikkal) kell felcserélni. Így elvileg minden tagot megmozgatunk, és jó átkeverés \square eredmény.

A fő paramétereket ezúttal is kívülről visszük be, tehát \square program univerzálisan használható.

Keverés

| | | | |
|-----|------|------|-----------|
| 10 | CIM | EQU | 23296 |
| 20 | RND | EQU | CIM+2 |
| 30 | NN | EQU | CIM+4 |
| 40 | TAG | EQU | CIM+6 |
| 50 | | ORG | 23310 |
| 60 | | LD | BC, (CIM) |
| 70 | | LD | DE, (RND) |
| 80 | | PUSH | BC |
| 90 | | LD | A, (NN) |
| 100 | | LD | B, A |
| 110 | CIKL | LD | A, B |
| 120 | | LD | (TAG), A |
| 130 | | POP | BC |
| 140 | | LD | A, (BC) |

| | | |
|-----|------|-----------|
| 150 | PUSH | AF |
| 160 | PUSH | BC |
| 170 | LD | A, (DE) |
| 180 | LD | C, A |
| 190 | LD | B, 0 |
| 200 | LD | HL, (CIM) |
| 210 | ADD | HL, BC |
| 220 | LD | A, (HL) |
| 230 | POP | BC |
| 240 | LD | (BC), A |
| 250 | POP | AF |
| 260 | LD | (HL), A |
| 270 | INC | DE |
| 280 | INC | BC |
| 290 | PUSH | BC |
| 300 | LD | A, (TAG) |
| 310 | LD | B, A |
| 320 | DJNZ | CIKL |
| 330 | RET | |

A program működése:

Az assembler direktívákkal kezdjük ismét \square beírást, ahol

- 10 - \square számsor kezdőcíme,
- 20 - \square véletlen \square kezdőcíme,
- 30 - \square sor tagjainak száma és
- 40 - \square ciklusváltozó tárolási helye.

A számsor kezdőcímét a BC-be, \square RND sor címét DE-be töltjük, majd \square kezdőcím verembe való kimentése után, \square tagok számát írjuk be \square már ismert módon a tárolás helyére (120).

Az aktuális címet a veremből vesszük ki (130), és \square cserélendő szám ott lévő értékét félretesszük \square verembe (150), ahová \square soros címet is visszatesszük (160).

A DE címen lévő véletlenszámot \square kezdőcímhöz kell hozzáadni; hogy ezt könnyen megtehessük, \square értéket egy megfelelő regiszterpárba kell tölteni, ami csak több lépésben lehetséges, majd \square összeadást elvégezve (210) HL-ben már \square csereszám címe van. Az innen kivett számértéket (220) a veremből kivett aktuális címre betöltjük (240), helyébe pedig azt \square értéket tesszük, amit előzőleg \square aktuális címről \square verembe mentettük (260).

A számsore végrehajtása után mindkét \square egyet lépünk előre, \square BC-ben lévő soron következő címet kimentjük \square verembe, mert \square BC regiszterpárt közben másra is használjuk. A ciklusváltozó lehívása (300) után, értékét csökkentjük, és amíg nem éri el \square 0-t, tovább folytatjuk \square műveletet.

A programozást kedvelő amatőrök közül kevesen tudnak ellenállni annak \square csábításnak, hogy elkészítsék \square közismert \square logikai feladvány programját, melynek \square lényege, hogy egy 4x4 mezőből álló táblán lévő 15 számot rendezetlen állapotból, tologatással, rendezett állapotba kell hozni. (Számítógépen megoldható \square más méretű, pl. 3x3 vagy 5x5 mezőből álló tábla előállítás is, ami \square játékot változatosabbá teszi).

A valamikor közkedvelt mechanikai felépítésű táblán (eredeti neve a feltaláló után "Lloyd") minden állásból rendbe lehetett hozni \square számokat, hiszen \square állás \square eredetileg rendezett állapotból, ugyancsak tologatással jött létre. ha azonban \square rendezetlen állapotot véletlenszerű keveréssel állítjuk elő, akkor matematikai törvényszerűség, hogy \square megoldhatóság valószínűsége \square %. Az amatőr programok nagy része - még \square egyébként meglepően színvonalasak is - hibás, ugyanis ezt a körülményt nem veszi figyelembe. A megoldás többek között \square Spencer: "Játékok BASIC nyelven" címmel magyarul is kiadott könyvben megtalálható. Az ellenőrző eljárás több műveletből áll:

1. A kevert számsor első tagjától kiindulva (majd ugyanezt minden egyes tagra megismételve) megszámloljuk, hogy hány nála kisebb értékű tag áll mögötte, és ezek számát összegezzük (az üres mezőt eredeti értékével, tehát pl. 4*4-es tábla esetén 16-tal kell figyelembe venni).
2. A táblát \square sakk táblához hasonlóan osszuk \square sötét és világos színű mezőkre. Ha \square üres mező színe nem egyezik meg \square jobb alsó sor színével, akkor \square előbb kapott értékhez még adjunk hozzá 1-et.

3. Ha \square így kapott érték páros, akkor \square tábla rendezhető. Egyébként két szomszédos számot (de nem \square üres mezőt!) kell egymással felcserélni, és \square számsor rendezhetővé válik.

Mintaprogramunk az 1. műveletre vonatkozik, aminek BASIC változata meglehetősen bonyolult és lassú, \square eljárás 2. és 3. pontjait \square játékprogramba kell beépíteni.

Kontroll

| | | | |
|-----|--------|------|--------------|
| 10 | CIM | EQU | 23296 |
| 20 | NN | EQU | CIM+2 |
| 30 | SORCIM | EQU | CIM+4 |
| 40 | SUMMA | EQU | CIM+6 |
| 50 | | ORG | 23310 |
| 60 | | LD | IX, SUMMA |
| 70 | | LD | (IX+0), 0 |
| 80 | | LD | HL, (CIM) |
| 90 | | LD | A, (NN) |
| 100 | | DEC | A |
| 110 | | LD | B, A |
| 120 | C1 | PUSH | BC |
| 130 | | LD | (SORCIM), HL |
| 140 | | LD | DE, (SORCIM) |
| 150 | C2 | INC | DE |
| 160 | | LD | A, (DE) |
| 170 | | CP | (HL) |
| 180 | | JR | NC, UGR |
| 190 | | INC | (IX+0) |
| 200 | UGR | DJNZ | C2 |
| 210 | | INC | HL |
| 220 | | POP | BC |
| 230 | | DJNZ | C1 |
| 240 | | RET | |

A program működése:

Az összegző memóriarekesz nullázása (70) után HL-be \square szám- \square kezdőcímét, A-ba a tagok számát töltjük. Ez utóbbinál 1-gyel kevesebb a szükséges lépések száma (100), és \square lesz a ciklus-változó, amit \square B regiszter hordoz (110).

A C1 ciklus első lépéseként a B-ben lévő ciklusváltozót mentjük ki \square verembe, majd \square HL-ben lévő soron következő címet töltjük \square tárolási helyére (130). Innen vesszük ki \square mindenkor soros cím értékét (140).

A C2 ciklus minden fordulóban egyet előre lépve (150) az ott talált értéket hasonlítja össze a soros, HL-ben lévő címen található bázisszámmal (170), és ha \square nagyobb értékű, akkor \square C jelzőbit értéke 1 lesz, és \square 180 sorból nincs ugrás, tehát \square számláló rekesz tartalma 1-gyel növekszik (190). A ciklus a hátralévő tagok számától függően ismétlődik, majd visszatérünk a C1 ciklusba, és \square következő tagra végezzük el \square vizsgálatot.

A végeredményt \square SUMMA nevű rekeszben találjuk, és \square már \square játékprogram dolga, hogy ezt hogyan használja fel.

Aki figyelmesen követte \square eddigi mintaprogramok gondolatmenetét, \square bátran vállalkozhat egy összetettebb feladat megoldására is, méghozzá \square eddigieknél szűkszavúbb magyarázatokra támaszkodva.

Legyen az új feladat egy olyan program, amely szöveges adatok (névsor, katalógus, címtár stb.) ABC sorrendbe való rendezésére alkalmas. Ennél \square témánál különösen szembevetendő \square BASIC és \square gépi kódú programok sebessége közötti különbség. A stringek legyenek egyforma hosszúak, és helyezkedjenek el \square memória adott címétől kezdve egymás után, egymástól \square távolságra. A gondolatmenet \square egyszerű számszortozó rutinhoz hasonló: Az első string első karakterét összehasonlítjuk \square mögötte állók első karakterével, és attól függően cserélünk vagy továbblépünk, hogy \square két kódszám közül melyik nagyobb. Egyenlőség esetén \square adott stringek első, második, harmadik stb. karaktereinek összehasonlítását kell elvégezni.

A program készítésénél vegyük figyelembe, hogy az ilyen jellegű listák sorai (többnyire 32 karakter soronként) általában sorszámból, címből és \square ezt követő jellemző adatokból állnak. Rende-

zéskor \square sorszám \square helyén marad, és csak a címet kell sorba rendezni, de csere esetén \square címmel együtt mozgathatjuk \square jellemző adatokat is.

Szórendező

| | | | |
|--|------|------|------------|
| 10 | NN | EQU | 65001 |
| ; Tagok számának tárolási helye | | | |
| 20 | KEZD | EQU | NN+2 |
| ; Adatblokk kezdőcím tároló helye | | | |
| 30 | | EQU | NN+4 |
| ; Egy teljes sor hossza | | | |
| 40 | CIM | EQU | NN+6 |
| ; Rendezendő cím hossza | | | |
| 50 | SZAM | EQU | NN+8 |
| ; Cím előtti sorszám hossza | | | |
| 60 | TAG | EQU | NN+10 |
| ; Tagok csökkenő száma (C1 ciklus) | | | |
| 70 | KCIM | EQU | NN+12 |
| ; Aktuális kezdőcím | | | |
| 80 | SCIM | EQU | NN+14 |
| ; Aktuális sorcím | | | |
| 90 | STAG | EQU | NN+16 |
| ; Soros tagok csökkenő száma (C2 cikl.) | | | |
| 100 | BETU | EQU | NN+18 |
| ; Tag betűinek változó száma (C3 cikl.) | | | |
| 110 | KAR | EQU | NN+20 |
| ; Cserélendő karakterek cikl. száma (C4) | | | |
| 120 | | ORG | 65100 |
| 130 | | EXX | |
| ; Nem árt \square óvatosság. HL értékének | | | |
| 140 | | PUSH | HL |
| ; megőrzésével \square BASIC-be való sikeres | | | |
| 150 | | EXX | |
| ; visszatérést biztosítjuk. | | | |
| 160 | | LD | HL, (KEZD) |
| ; Adatblokk kezdőcíme HL-ben | | | |
| 170 | | LD | DE, (SZAM) |
| ; Sorszám hossza DE-ben | | | |
| 180 | | ADD | HL, DE |
| ; Sorszám átugrása, | | | |
| 190 | | LD | (KCIM), HL |
| ; és itt \square aktuális kezdőcím. | | | |
| 200 | | LD | BC, (NN) |
| ; Tagok száma BC-ben, | | | |
| 210 | | DEC | BC |
| ; de csak NN-1 lépés kell. | | | |
| 220 | C1 | LD | (TAG), BC |
| ; Lépésszám \square ciklusváltozó | | | |
| 230 | | LD | HL, (KCIM) |
| ; Aktuális kezdőcím HL-ben | | | |
| 240 | | LD | (SCIM), HL |
| ; Ez lesz \square induló sorcím is. | | | |
| 250 | C2 | LD | (STAG), BC |
| ; C2 ciklusváltozó \square helyén | | | |
| 260 | | LD | HL, (KCIM) |
| ; Aktuális kezdőcím HL-ben, | | | |
| 270 | | LD | A, (HL) |
| ; és ennek tartalma A-ban. | | | |
| 280 | | LD | HL, (SCIM) |
| ; Aktuális sorcím HL-ben | | | |
| 290 | | LD | DE, (MM) |
| ; Sor hossza lesz \square lépéseköz | | | |
| 300 | | | HL, DE |
| ; Ugrás \square következő címre | | | |
| 310 | | LD | (SCIM), HL |
| ; Az új sorcímet visszatöltjük, | | | |
| 320 | | CP | (HL) |
| ; tartalmát A-val hasonlítjuk, | | | |
| 330 | | JR | NZ, U1 |
| ; és ugrás, ha A nem egyenlő (HL)-el | | | |
| 340 | | CALL | BELSO |
| ; A további karakterek vizsgálata | | | |
| 350 | U1 | CP | (HL) |
| ; ismételt összehasonlítás | | | |


```

360      JR    C,U2
; és ugrás, ha (HL) > A
370      CALL CSERE
; Egyébként szócsera meghívása
380 U2      LD    BC,(STAG)
; BC-ben a C2 ciklusváltozója
390      DEC    BC
400      LD    A,B
410      OR     C
; Megjegyzés a program végén!
420      JR    NZ,C2
; Vissza az elejére, amíg BC nem egyenlő
; (0)-val. Ha nincs tovább, a C1 ciklus
; folytatható.
430      LD    HL,(KCIM)
; Aktuális kezdőcím
440      LD    DE,(MM)
; és lépésköz lehívása
450      ADD    HL,DE
; Így kapjuk a következő kezdőcímet,
460      LD    (KCIM),HL
; amit a tároló rekeszben megőrzünk.
470      LD    BC,(TAG)
; C1 ciklusváltozó lehívása
480      DEC    BC
490      LD    A,B
500      OR     C
510      JR    NZ,C1
; Vissza amíg BC nem egyenlő (0)-val,
; egyébként a program befejezhető.
520      EXX    HL
530      POP    HL
540      EXX    HL
550      RET
560 BELSO   LD    DE,(KCIM)
; További karakterek
; összehasonlításához
570      LD    HL,(SCIM)
; az indulási adatok
580      LD    A,(CIM)
; lehívása.
590      DEC    A
; Csak A-1 lépést kell tenni.
600      LD    B,A
610 C3      LD    A,B
; Ez a ciklusváltozó,
620      LD    (BETU),A
; amit megőrzünk
630      INC    DE
; Következő karakter helye
640      INC    HL
; mindkét címben
650      LD    A,(DE)
; és az ott talált értéket
660      CP    (HL)
; összehasonlítása.
670      JR    NZ,U3
; Ugrás, ha A nem egyenlő (HL)-lel

```

```

680      LD    A,(BETU)
; Lehívjuk a C3 ciklusváltozót
690      LD    B,A
700      DJNZ  C3
; B-1, és vissza ha B nem egyenlő 0-val,
710      RET
; egyébként vége.
720 CSERE   LD    DE,(KCIM)
; Szócsera szubrutin induló adatainak
730      LD    HL,(SCIM)
; betöltése.
740      LD    A,(SZAM)
750      LD    B,A
760      LD    A,(MM)
; A-ban egy teljes sor hossza
770      SUB    B
; amiből a sorszám hosszát kivonjuk
780      LD    B,A
; és ebből lesz B-ben a
790 C4      LD    A,B
; C4 ciklusváltozó,
800      LD    (KAR),A
; amit megőrzünk
810      LD    A,(DE)
; DE-ben még a KCIM tartalma van,
820      LD    B,A
830      LD    A,(HL)
; HL-ben pedig az SCIM-e
840      LD    (DE),A
; és a két címen levő értékeket
850      LD    (HL),B
; egymással felcseréljük.
860      INC    HL
; Következő karakter címe
870      INC    DE
; mindkét címben.
880      LD    A,(KAR)
; C4 ciklusváltozó
890      LD    B,A
900      DJNZ  C4
; B-1, és vissza, amíg B nem egyenlő 0-val
910      RET
; egyébként a szubrutin vége.

```

Megjegyzés: a 390...420, továbbá a 470...510 utasítások során lényegében a BC regiszterpárban hordozott ciklusváltozó értékét csökkentjük, majd ellenőrizzük, hogy az elérte-e már a 0 értéket (tehát B=0 és C=0). Az itt alkalmazott frappáns megoldás az OR logikai utasításának azt a sajátosságát használja ki, hogy annak elvégzése után (410) az A regiszter értéke csak akkor lesz 0, ha előzőleg mindkét változó – vagyis az A-ba töltött B, valamint a C értéke 0 volt.

A program, természetesen, bárhol elhelyezhető, akár az addigiekhez hasonlóan a nyomtató puffertérületén, de ne feledjük, hogy az ilyen jellegű programokhoz gyakran használnak sornyomtatót. A 10..50 adatokat kívülről kell bevinni, ami az univerzális alkalmazást teszi lehetővé.

Painter

A POKE-olást a BASIC/22800/25 file térképpel rendelkező verzióra fogjuk közölni.

MERGE "-dzzseljük be a BASIC betöltőt, majd állítsuk meg a magnót, és várjunk kb. 9 másodpercet, türelmesen. Ekkor megjelenik az OK. üzenet. A 2. sorból töröljük ki a 2 db. POKE utasítást, majd EDIT-eljük a 620-as sort:

LIST 620 (ENTER), CAPS SHIFT + 1

A 620-as sorban a „liv” érték adja meg az életek számát, ide írhatunk bármennyit, pl. 500, vagy 1-et is.

Adjuk ki: RUN (ENTER), majd töltsük be a 2 kódot, és annyi életünk lesz, amennyit a 620-as sorba beírtunk.

Gyakorlatilag az örökéletet is hasonló módszerrel érhetjük el, annyi különbséggel, hogy a 620-as sor EDIT-álása helyett a 2020-as sort kell kitörölnünk.

Tartalomjegyzék

| | | |
|-----|---|----|
| 1 | Bevezetés helyett | 1 |
| 2. | Játék, POKE, térkép | 2 |
| | - SOLDIER OF FORTUNE, NETHERWORLD | 2 |
| | - FERNANDEZ MUST DIE... | 3 |
| | - ...SIR FRED | 4 |
| | - R-TYPE | 5 |
| 2.1 | Heavy On the Magick! (Gargoyle Games) | 8 |
| 3. | ENTERFACE (Enterprise melléklet) | 15 |
| 4. | Ismeretlen nyelvek (Micro-PROLOG: A perifériák kezelése) | 19 |
| | - KILLED UNTIL DEAD (Level III.) | 21 |
| 5. | Ismeretlen nyelvek (HISOFT 'C' Compiler) | 22 |
| 6. | Hardware ötletek (Numerikus billentyűzet) | 25 |
| | - KILLED UNTIL DEAD (Level IV.) | 26 |
| 7. | BASIC (A billentyűzet figyelése, grafikai ötletek) | 27 |
| 8. | Gépi kód tanfolyam | 28 |



SpV. 18.rész EP melléklet Mercenary

- Az űrsiklót nem szükséges megvenni, el is lophatjuk. Ekkor nagy valószínűséggel lelőnek, de sebj, a CAPS SHIFT + Q-val új űrhajót kérhetünk.
- Az űrállomáson lévő halálfejes ajtó nem halálos, csak lezuhanunk, és hosszú sétát kell tennünk egy hangárig, melyben van űrsikló, vagy segít a CAPS SHIFT + Q.
- A célkereszt (sights) nagymértékben megkönnyíti az ajtókon való bejutást.
- A 03-15 szektorba könnyű eljutni, ha a 09-06 szektorban az orvosi felszerelésektől kimegyünk a folyosóra, és bemegyünk a szemközti ajtón. A második ajtó balra, és ha szerencsénk van, akkor a 03-15-ben kötünk ki, ha nem, akkor próbálkozunk tovább.

SpV. 14.rész 5. oldal Garfield

A sintértelep kulcsának megszerzésére van egy egyszerűbb módszer is: Menjünk el Nermál barátunkhoz, aki a csatornában van az egérrel. (Ha a csatornába leviszük a lámpát, az világít nekünk.) Nermált kb. úgy jó 5-6-szor rúgjuk fenékre, ekkor leejti a felhúzható egeret. Ezt kapjuk fel, és vigyük ki a csatornarendszerből. Ha tehát Nermál keze üres, rohagál a csatornarendszerben. Álljunk készen a láda melletti szobában, majd amikor megjelenik Nermál, induljunk el a ládához. Ahogy a ládához értünk, rúgjunk bele a ládába, ő felveszi a csontot, mi pedig felvehetjük a kulcsot. A gaz patkány, aki a ládát őrzi, nem tud beleszólni az akcióba.

SpV. 18.rész, EP.melléklet SPECTRUM programok átírása

A programlistába két helyen is hiba csúszott:

| | | | | | |
|----|------|-----------|-----------|-----|------------|
| L1 | LD | (HL),A | LD FLAG | RRA | és nem RRC |
| | INC | HL | | XOR | L |
| | LD | A,(3FFFh) | | RET | NZ |
| | CALL | WRA | és nem WR | | |

A hibák kijavítása után a program már jól fut.

A MŰSZAKI KÖNYVKIADÓ könyvvajánlata
A felsorolt könyvek megrendelhetők, ill. megvásárolhatók:
Műszaki Könyvkiadó Kandó Kálmán könyvesboltja,
Budapest V., Bajcsy-Zs. út 20. 1051

Könöczy - Gál

Csupa szuperjáték C-PLUS/4, C-16, C-116

Kb. 156 oldal, 135 Ft

A könyv a Csupa játék sorozat legújabb kötete, amelyben játékok és felhasználói programok, valamint "tippek és trükkök" találhatók. A játékok látványos grafikával és kellemes zenével készülnek, jó szórakozást ígérnek. A felhasználói programok nagy segítséget nyújtanak a programíráshoz, a grafika előállításához, ill. programok másolásához. A programok magas színvonalúak, remélhetően sok örömet szereznek a gyakorlott és kezdő játékosok, valamint egyéb felhasználók számára.

Mazzag Mihály

LOTUS 1-2-3

(lapozgató sorozat)

Kb. 240 oldal, 220 Ft

A táblázatkezelő programok elterjesztésének és tényleges használatának gyorsítására mindent el kell követni, mert a felhasználó számára a számítástechnika használata csak a programok teszik legközvetlenebbül hozzáférhetővé. A LOTUS 1-2-3 az egyik legismertebb táblázatkezelő program, de csak elvétve találkozhatunk magyar nyelvű leírásával. A lapozgató sorozat, amely az IBM PC-hez kapcsolódó programok tömör, pontos leírása példákkal és megjegyzésekkel kiegészítve, következő kötetével ezt a hiányt akarja pótolni.

A könyv mintegy 150 parancssorozatot, 80 függvényt és 50 makrót ismertet. Mindegyiknél megadja a formátumot, a verziószámot és alkalmazási lehetőségeit közöl. A lapozgató sorozat nem pótolja a dokumentációt, de olyan segédeszköz, amely minden felhasználónak gyorsan megtalálható, pontos információt ad.

Magyar István

OPEN ACCESS

(lapozgató sorozat)

Kb. 200 oldal, 180 Ft

Az OPEN ACCESS integrált szoftver a kevés számítástechnikai gyakorlattal rendelkező felhasználói kör számára olyan eszköz, amely széles körű szolgáltatásaival (adatbáziskezelés, számológéptábla, szövegfeldolgozás, kalendárium, grafika, kommunikáció) képes az alkalmazói igények kielégítésére, kezelése pedig gyorsan és egyszerűen elsajátítható.

A lapozgató sorozat e kötete az alapvető tudnivalók összefoglalása után tömören, a formai előírások pontos betartásával ismerteti a programmodulok működését és a parancskészlet használatát. Külön fejezet foglalkozik a hibáüzenetekkel.

Kőhegyi János

Ismerd meg a BASIC nyelvjárásait!

(Commodore-16, Commodore PLUS/4, Commodore-128, Videoton TV-Computer)

Kb. 144 oldal, 135 Ft

A BASIC nyelv a számítógépet használók körében közzismert nyelv, amelynek géptípusonként sajátos változatai vannak. A könyv Donald Alcock: Ismerd meg a BASIC nyelvet c. közzismert könyvére támaszkodva a Commodore-16, Commodore PLUS/4, Commodore-128 és Videoton TV-Computer BASIC nyelvjárásait ismerteti.

Tartalom:

Előszó / C16 BASIC / Commodore PLUS/4 BASIC / Commodore 128 BASIC / Videoton TV Computer BASIC

A 'SpV' 20. számában megjelent kérésrejtélyes helyes megfejtései:

Válasz: EGYRI KEVÉSSEBHEN, 76. TÖBLIAS KALANDJAI, Füg. 1. ESPIONAGE ISLAND, 17. LE KELI FORDÍTANT

A 19. szám megrendelés a nyertesek: HLA - Budapest XII (S31), H.J. - Halászfalok (C101), K.L. - Kerepestarcza (S126), K.T. - Szigetszentmiklós (S95), V.B. - Budapest XXII (S136), nyerteseműket postázta!